Regular sets over extended tree structures

Severine Fratani

Regular sets over extended tree structures

Séverine Fratani

Laboratoire d'Informatique Fondamentale de Marseille (LIF) CNRS : UMR6166
Université de la Méditerranée - Aix-Marseille II
Université de Provence - Aix-Marseille I

Abstract

We investigate notions of decidability and definability for the Monadic Second-Order Logic of labeled tree structures, and links with finite automata using oracles to test input prefixes.

A general framework is defined allowing to transfer some MSO-properties from a graph-structure to a labeled tree structure. Transferred properties are decidability of sentences and existence of a definable model for every satisfiable formula. A class of finite automata with prefix-oracles is also defined, recognizing exactly languages defined by MSO-formulas in any labeled tree-structure.

Applying these results, the well-known equality between languages recognized by finite automata, sets of vertices MSO definable in a tree-structure and sets of pushdown contexts generated by pushdown-automata is extended to k-iterated pushdown automata.

Key words: Labeled tree structures; MSO definable sets; Automata with oracle; Iterated pushdown structures.

Introduction

Initiated by the work of Büchi on words, the study of links between automata and logic has permit to identify structures having a decidable Monadic Second-Order theory. In particular, Rabin proved in [28] decidability of the MSO-theory of infinite tree structures in which numerous properties are definable and theories are interpretable. These works have also led to a logic characterisation of regular languages: languages recognised by finite automata are exactly sets defined by MSO-formulas in a tree structure.

The goal of this paper is to extend these works to the study of labelled tree structures: identify labellings for which tree structure have a decidable MSO-theory, for which every formula admits a definable model and give an automata-characterization of the definable sets.

To achieve this goal, we introduce new interesting objects and results. First, we define a class of word/tree automata with prefix-oracles (i.e., sets of words over the input alphabet) used to test the already processed prefixes of inputs. Languages and forest recognized by prefix-oracles automata enjoy nice property,

in particular, the Rabin's correspondence between regular forests and models of MSO-formulas over infinite trees can be extended to these languages: forests recognized by automata with oracles $O_1, \ldots, O_m$ are forests MSO-definable in tree structures extended by unary relations $O_1, \ldots, O_m$. Remark that this approach has already been devised in [32] to characterise some proper subclasses of regular languages by using regular prefix-oracles and to study their definability in First-Order Logic over extended word structures. However, the definition of automata with prefix-oracles does not explicitly appear in this paper since regular prefix-oracles can be simulated by the direct product of finite automata. Second, we establish transfer theorems, allowing from a structure, to construct a tree structure having some similar MSO properties. This approach is common for the transfer of decidability (for example the transfer of decidability from a structure to its *tree-like structure*, (see [31] or [35]), or from a graph to it's unfolding (see [10])), but here, in addition to decidability, transferred properties also applied to sets *MSO definable* in such structures and classes of automata *recognising* them. In addition, our transfer of decidability allows to obtain new decidability results which are not cover by the ones cited above. Properties are transferred to a labelled tree structure from its *image structure* by any morphism. If $\mu : D \to D'$ is a mapping and $\mathcal{S}$ is a relational structure over D, the *image structure* $\mu(\mathcal{S})$ of $\mathcal{S}$ has D' as domain and its relations are the images by μ of the relations of $\mathcal{S}$.

Let t be a labelled tree, and $\underline{t}$ be the structure associated to t. For any morphism of monoid μ, and under some simple hypothesis on the labelling of t, we obtain the following main results:

- *Transfer of decidability:* (Theorem 55) if $\mu(\underline{t})$ has a decidable MSO-theory, then $\underline{t}$ has a decidable MSO-theory,
- *Transfer of the property of Definable Model:* (Theorem 57) under a condition on μ, if $\mu(\underline{t})$ satisfies the property of *Definable Model* (DM), then $\underline{t}$ satisfies DM. This property ensures for a structure $\mathcal{S}$ that any satisfiable formula admits at least one model which is MSO-definable in $\mathcal{S}$ (see Definition 15),
- *Theorem of structure:* (Theorem 58) under the same condition on μ, if $\mu(\underline{t})$ satisfies DM, then any set is MSO-definable in $\underline{t}$ iff it is recognised by a finite automaton using only oracles of the form $\mu^{-1}(D)$ where D is MSO-definable in $\mu(\underline{t})$. (Then each oracle tests a property MSO-definable in $\mu(\underline{t})$, on the image by μ of input word prefixes).

Applying these results, we obtain tree structures having a decidable MSO theory and classes of languages having two equivalent characterizations: as languages recognized by automata with oracles, and as sets MSO-definable in some labelled tree structures. We thus extend the two characterizations of regular languages mentioned above.

But regular languages admit a third characterization, as sets of pushdown contexts generated by a pushdown system of transitions [21]. Some recent works

deal with "iterated pushdown automaton", which are automata whose memory is roughly a stack of stack ... of stack (see for examples [5, 7, 24, 19]), it is then natural to attempt to define a notion of "regular" sets of k-pushdowns (i.e., stacks with k level of embedded pushdowns) which generalize the previous equality. We give equalities between languages of k-pushdowns recognized by automata with p-oracles, languages MSO-definable in a particular tree structure and sets of k-pushdown contexts generated by a k-pushdown system of transitions. We iteratively use the three transfer theorems on a family of structures $(\boldsymbol{\mathcal{P}_k})_{k\geq 1}$ having a prefix words language $\mathcal{P}_k$ for domain. The language $\mathcal{P}_k$ defines an encoding of the set k-pds of all k-pushdowns. The structure $\boldsymbol{\mathcal{P}_k}$ is MSO-*equivalent* with the structure PDS_k whose domain is k-pds and whose relations are those induced by the classical instructions on k-pushdowns. This allows to define a class of languages in $\mathcal{P}_k$ that can be expressed in four equivalent ways (Theorem 85):

- as languages recognised by finite automata with prefix-oracle,
- as languages defined by MSO-formulas in the tree structure of domain $\mathcal{P}_k$,
- as encodings of sets defined by MSO-formulas in PDS_k,
- as encoding of sets of k-pushdowns generated by a store-controlled k-pds system of transitions.

We show in addition that PDS_k satisfies the property of Definable Model.

This paper is organized as follows. Section 1 is devoted to basic definitions on words, logic, automata and k-pushdowns structures. It is also introduced the notion of word automata with oracles. In Section 2, we extend to tree automata the use of oracles. The Rabin's correspondence between regular forests and models of MSO-formulas over trees is adapted to these languages. In Section 3, we develop a game-theoretical approach to prove the three transfer theorems. We give also a simple application of the transfer theorems. Finally, we give in Section 4 a definition of k-regular sets of pushdowns.

1. Preliminaries

1.1. Basic definitions

1.1.1. Some notations and conventions

Given a set A, we denote by $|A|$ the cardinal of A. If s is a map from a set A, then $s(A) = \{s(a) \mid a \in A\}$. If $\vec{V} = (V_1, \ldots, V_n)$ is a vector of subsets of A then $s(\vec{A}) = (s(V_1), \ldots, s(V_n))$. The characteristic function of $\vec{V}$ in A is a map $\chi_A^{\vec{V}} : A \to \{0,1\}^n$ defined for all $x \in A$, by $\chi_A^{\vec{V}}(x) = (b_1, \ldots, b_n)$ where $\forall i$, $b_i = 1$ iff $u \in S_i$.

1.1.2. Words and languages

If A is a set, A^* denotes the set of words (finite sequences) over A, and ε the empty word. For $u, v \in A^*$, the length of u is denoted $|u|$ and we write $v \preccurlyeq u$ if v is prefix of u. A set $P \subseteq A^*$ is a prefix closed language if $\forall u \in P$, $\forall v \in A^*$, if $v \preccurlyeq u$ then $v \in P$.

1.1.3. Free group

Given a finite alphabet A , let us associate to each $a \in A$ the inverse symbol $\bar{a}$ which does not belong to A. We denote by $\overline{A}$ the set of inverse letters of A and define $\widehat{A} = A \cup \overline{A}$. For every $u = a_1 \cdots a_n \in \widehat{A}^*$, the **inverse** word of u is $\overline{u} = b_n \cdots b_1$ where $\forall i \in [1, n]$:

$$\text{if } a_i \in A \text{ then } b_i = \bar{a_i}, \text{ and if } a_i = \bar{a} \in \overline{A} \text{ then } b_i = a.$$

Let us then consider the reduction system $S = \{(a\bar{a}, \varepsilon), (\bar{a}a, \varepsilon)\}$. A word in $\widehat{A}^*$ is said to be **reduced** if it is S-reduced, i.e., it does not contain occurrences of $a\bar{a}$ or $\bar{a}a$, for $a \in A$. We denote by $\mathrm{Irr}(A)$ the set of reduced words in $\widehat{A}^*$. As S is confluent, each word w is equivalent (mod $\leftrightarrow_S^*$) to a unique reduced word denoted $\rho(w)$.
We define the free group $(\mathrm{Irr}(A), \varepsilon, \bullet)$, where $\forall u, v \in \mathrm{Irr}(A)$, $u \bullet v = \rho(u \cdot v)$.

1.1.4. Projections

For any integers $0 < i \leq j \leq n$, for any vector of elements $(a_1, \ldots, a_n)$, we define the projection $\pi_i(a_1, \ldots, a_n) = a_i$ and $\pi_{i,\ldots,j}(a_1, \ldots, a_n) = (a_i, \ldots, a_j)$. For any alphabets B and A with $B \subseteq A$, the projection $\pi_B : A^* \to B^*$ is a morphism defined $\forall a \in A$ by $\pi_B(a) = a$ if $a \in B$ and $\pi_B(a) = \varepsilon$ else.

1.1.5. Trees and forests

Given finite alphabets Σ and A and a prefix closed language $P \subseteq A^*$, a P-tree(Σ) (tree of domain P labelled by Σ) is a total function $t : P \to \Sigma$. The set of all P-tree(Σ) is denoted P-$\mathsf{Tree}(\Sigma)$. In order to deal with unlabelled trees in an uniform way, we introduce the special symbol $\top$. Unlabelled trees are then functions $t : P \to \{\top\}$. We will often consider trees in P-$\mathsf{Tree}(\{0,1\}^n)$, for $n \geq 0$ (with the convention that $\{0,1\}^0 = \{\top\}$), we will denote this class P-Tree_n. Remark that a tree in P-Tree_n can always be seen as the characteristic function $\chi_P^{\vec{S}}$ of a vector $\vec{S} = (S_1, \ldots, S_n)$, for $S_i \subseteq P$.

We will use two kinds of operations on trees and tree-languages:

- Restriction: let $t \in A^*$-$\mathsf{Tree}(\Sigma)$, $t_{|P}$ is the P-tree(Σ) obtained by restricting the domain of t to P. If $F \subseteq A^*$-$\mathsf{Tree}(\Sigma)$, then $F_{|P} = \{t_{|P},\ t \in F\}$.

- Product: let t_1 be a P-tree(Σ_1) and t_2 a P-tree(Σ_2), the product of t_1 and t_2 is the tree $t_1{\hat{}}t_2 \in P$-$\mathsf{Tree}(\Sigma_1 \times \Sigma_2)$ fulfilling $\forall u \in P$, $t_1{\hat{}}t_2(u) = (t_1(u), t_2(u))$. This definition can be extended to tree languages:
  if F_1, $F_2 \subseteq P$-$\mathsf{Tree}(\Sigma)$, then $F_1{\hat{}}F_2 = \{t_1{\hat{}}t_2 |\ t_1 \in F_1,\ t_2 \in F_2\}$.

1.2. Finite automata with prefix-oracle

Finite automata with prefix-oracle (or p-oracle) extend the class of finite automata by allowing some membership tests on prefix of the input word. An automaton $\mathcal{A}$ with p-oracles, on the input alphabet A is a finite automaton associated to a vector $\vec{O} = (O_1, \ldots, O_m)$ of subsets of A^* and whose transitions contain a boolean vector of size m called **test**. During the computation by $\mathcal{A}$ of an input word, the already processed part u of the input is kept in memory and a transition with test $\vec{o}$ can be applied if $\vec{o}$ is equal to the characteristic vector of u inside $\vec{O}$ (i.e., if $\vec{o} = \chi^{\vec{O}}_{A^*}(u)$).

Definition 1 (Finite automaton with p-oracles). Given $m \geq 1$, an automaton with m p-oracles is a tuple $\mathcal{A} = (Q, A, \vec{O}, \Delta, q_0, F)$ where Q is a finite set of states, A is the input alphabet, $\vec{O} = (O_1, \ldots, O_m)$, $O_i \subseteq A^*$, $\Delta \subseteq Q \times A \times \{0,1\}^m \times Q$ is the set of transitions, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is the set of final states.
A configuration of $\mathcal{A}$ is a pair $(q, u_\uparrow v)$ where $uv \in A^*$ and $\uparrow$ is a symbol which does not belong to A. The binary relation on configurations is $\rightarrow_{\mathcal{A}}$ and consists in all pairs $(q, u_\uparrow av) \rightarrow_{\mathcal{A}} (p, ua_\uparrow v)$ such that $(q, a, \chi^{\vec{O}}_{A^*}(u), p) \in \Delta$. The language recognised by $\mathcal{A}$ is $\mathrm{L}(\mathcal{A}) = \{u \in A^* \mid (q_0, {}_\uparrow u) \rightarrow^*_{\mathcal{A}} (q_F, u_\uparrow),\ q_F \in F\}$.

We will use the following notations: $\mathsf{FA}^{\vec{O}}(A)$ is the family of automata over A with p-oracle $\vec{O}$, the class of $\vec{O}$-regular languages (i.e., recognised by automata in $\mathsf{FA}^{\vec{O}}(A)$) is $\mathsf{REG}^{\vec{O}}(A)$. Remark that an automaton with oracle $\emptyset$ is simply a finite automaton. We write then FA rather than $\mathsf{FA}^{\emptyset}$ and REG for $\mathsf{REG}^{\emptyset}$.

Definition 2. An automaton with m p-oracles $\mathcal{A} = (Q, A, \vec{O}, \Delta, q_0, F)$ is said to be deterministic if $\forall p \in Q$, $a \in A$, $\vec{o} \in \{0,1\}^m$, there is one and only one $q \in Q$ such that $(p, a, \vec{o}, q) \in \Delta$.

Example 3. The following automaton is deterministic and recognize the language $\{a^n b^n c^n\}_{n \geq 1}$.
$\mathcal{A} = (\{q_0, q_1, q_2, q_3, q_F\}, \{a, b, c\}, (O_1, O_2), \Delta, q_0, \{q_F\})$ where $O_1 = \{a^n b^n\}_{n \geq 1}$, $O_2 = \{a^m b^n c^{n-1}\}_{n \geq 1,\, m \geq 0}$ and Δ consists in
$(q_0, a, (0,0), q_1)$, $(q_1, a, (0,0), q_1)$, $(q_1, b, (0,0), q_2)$,
$(q_2, b, (0,0), q_2)$, $(q_2, b, (0,1), q_2)$ $(q_2, c, (1,0), q_3)$, $(q_2, c, (1,1), q_F)$,
$(q_3, c, (0,0), q_3)$ and $(q_3, c, (0,1), q_F)$.

We associate with each automaton with m p-oracles $\mathcal{A} \in \mathsf{FA}^{\vec{O}}(A)$, a finite automaton $\widetilde{\mathcal{A}} \in \mathsf{FA}(A \times \{0,1\}^m)$ called **source** of $\mathcal{A}$ and constructed by moving the test of each transition into the input letter of the transition: each transition $(p, a, \vec{o}, q)$ is transformed in $(p, (a, \vec{o}), q)$. The language $\mathrm{L}(\mathcal{A})$ can be obtain from the language $\mathrm{L}(\widetilde{\mathcal{A}})$ and the "characteristic language of $\vec{O}$ in A.

Definition 4. For every $\vec{O} = (O_1, \ldots, O_m)$, $O_i \subseteq A^*$, the *characteristic language* of $\vec{O}$ is defined by:
$\mathrm{L}^{\vec{O}}_{\chi} = \{(a_1, \vec{o}_1) \ldots (a_n, \vec{o}_n) \in (A \times \{0,1\}^m)^* \mid \forall i \in [1, n],\ \vec{o}_i = \chi^{\vec{O}}_{A^*}(a_1 \ldots a_{i-1})\}$.

Observation 5. For every $\vec{O} = (O_1, \ldots, O_m)$, $O_i \subseteq A^*$:

$$\mathsf{REG}^{\vec{O}}(A) = \{\pi_1(L \cap \mathrm{L}_\chi^{\vec{O}}) \mid L \in \mathsf{REG}(A \times \{0,1\}^m)\}.$$

Using the Kleene's theorem, we obtain easily:

Theorem 6. *Let A an alphabet, and $\vec{O}$ a vector of subsets of A^*,*

1. $\mathsf{REG}^{\vec{O}}(A)$ *is the class of languages recognized by deterministic automata in* $\mathsf{FA}^{\vec{O}}(A)$,
2. $\mathsf{REG}^{\vec{O}}(A)$ *is closed under boolean operations.*

1.3. Iterated pushdown stores

Originally defined by Greibach in [22], iterated pushdown stores are storage structures built iteratively. Let us fix an infinite sequence $\mathcal{A} = A_1, A_2, \ldots, A_k, \ldots$ of disjoint and finite alphabets. For all $k \geq 1$, we denote by $\mathcal{A}_k$ the finite sequence $A_1, \ldots, A_k$ and adopt the convention that $\mathcal{A}_0 = \{\bot\}$ and that $\mathcal{A}_0 \cap A_i$ is empty of all $i \geq 1$.

Definition 7. We define inductively the set k-$\mathsf{pds}(\mathcal{A}_k)$ (or k-pds when alphabets of store are understood) of k-iterated pushdown-stores over $\mathcal{A}_k$:
0-$\mathsf{pds}(\mathcal{A}_0) = \{\bot\}$,
$(k+1)$-$\mathsf{pds}(\mathcal{A}_{k+1}) = (A_{k+1}[k\text{-}\mathsf{pds}(\mathcal{A}_k)])^* \bot [k\text{-}\mathsf{pds}(\mathcal{A}_k)]$.

The set for all k-pushdowns for $k \geq 0$ is denoted it-$\mathsf{pds}(\mathcal{A})$. In the rest of the paper, any 1-pds $a_1[\bot]a_2[\bot]\cdots a_n[\bot] \bot [\bot]$ will be written simply $a_1 \ldots a_n \bot$ and $\forall k \geq 0$. We denote by $\bot_k$ the "empty" k-pds containing only symbols $\bot$: $\bot_0 = \bot$ and $\bot_{k+1} = \bot [\bot_k]$.

From the definition, every ω in $(k+1)$-$\mathsf{pds}(\mathcal{A}_{k+1})$, $k \geq 0$, has a unique decomposition as $\omega = a[\omega_1]\omega'$ with $\omega_1 \in k$-$\mathsf{pds}(A_k)$, $\omega' \in (k+1)(A_{k+1})$-$\mathsf{pds} \cup \{\varepsilon\}$ and $a \in A_{k+1} \cup \{\bot\}$. Furthermore, $a = \bot$ iff $\omega' = \varepsilon$.

Example 8. Let $A_1 = \{a_1, b_1\}$, $A_2 = \{a_2, b_2\}$, $A_3 = \{a_3\}$ be storage alphabets, $\omega_{ex} = a_3[b_2[b_1 a_1 \bot]a_2[a_1 \bot] \bot_2] \bot [a_2[a_1 \bot]a_2[\bot] \bot_2] \in 3$-$\mathsf{pds}(\mathcal{A}_3)$.
Its decomposition corresponds to $a = a_3$, $\omega_1 = b_2[b_1 a_1 \bot]a_2[a_1 \bot] \bot_2$ and $\omega' = \bot [a_2[a_1 \bot]a_2[\bot] \bot_2]$.

The two following maps will be useful.

Projection: the map associating each k-pds to its top i-pds, $1 \leq i \leq k$ is $\mathrm{p}_{k,i}$: k-$\mathsf{pds}(A_1, \ldots, A_k) \to i$-$\mathsf{pds}(A_1, \ldots, A_i)$, where $\forall \omega = a[\omega_1]\omega' \in k$-$\mathsf{pds}$,

$$\mathrm{p}_{k,k}(\omega) = \omega \text{ and } \mathrm{p}_{k,i}(\omega) = \mathrm{p}_{k-1,i}(\omega_1) \text{ if } 1 \leq i \leq k-1.$$

The double subscript notation will be used to handle inverse functions, the rest of the time, we will note p_i for $\mathrm{p}_{k,i}$.

Top symbols: the map associating any k-pds, $k \geq 1$ to its k top-symbols is top : k-$\mathsf{pds}(A_1, \ldots, A_k) \to (A_k \cup \{\bot\}) \cdots (A_2 \cup \{\bot\})(A_1 \cup \{\bot\})$ defined $\forall \omega = a[\omega_1]\omega' \in k$-$\mathsf{pds}$ by

$$\mathrm{top}(\omega) = a, \text{ if } k = 1, \text{ else } \mathrm{top}(\omega) = a \cdot \mathrm{top}(\omega_1).$$

For $i \in [1,k]$, and $\omega \in k\text{-}\mathsf{pds}$, we denote by $\mathrm{top}_i(\omega)$ the i-th letter of $\mathrm{top}(\omega)$.

Example 9. Let ω_{ex} be the 3-pds given in Example 8:
$\mathrm{p}_2(\omega_{ex}) = b_2[b_1a_1\ \bot]a_2[a_1\ \bot]\ \bot_2$, $\mathrm{p}_1(\omega_{ex}) = b_1a_1\ \bot$, and
$\mathrm{top}(\omega_{ex}) = a_3b_2b_1$, $\mathrm{top}(\mathrm{p}_2(\omega_{ex})) = b_2b_1$, $\mathrm{top}(\mathrm{p}_1(\omega_{ex})) = b_1$.

An **instruction** on it-pds is a function from it-pds to it-pds which does not modify the level of the k-pushdowns (i.e., if instr is an instruction then for any $k \geq 0$ and any $\omega \in k\text{-}\mathsf{pds}$, $\mathsf{instr}(\omega) \in k\text{-}\mathsf{pds}$). An **instruction of level** i is an instruction which does not modify the levels greater than i of any it-pds. Hence, given instr an instruction of level i

if $\omega = a[\omega_1]\omega' \in k\text{-}\mathsf{pds}$, $k > i$, then $\mathsf{instr}(\omega) = a[\mathsf{instr}(\omega_1)]\omega'$

if $\omega \in k\text{-}\mathsf{pds}$, $k < i$, then $\mathsf{instr}(w) = w$.

Therefore, to define an instruction of level i, there is only need to define it for any $\omega \in i\text{-}\mathsf{pds}$.

Three instructions of level k are generally applicable to it-pushdowns.

Definition 10. "Classical" instructions of level $i \geq 1$ over $\mathcal{A}$ are defined for every $\omega = b[\omega_1]\omega' \in i\text{-}\mathsf{pds}(\mathcal{A}_i)$ by:

$\mathrm{pop}_i(\omega) = \omega'$ if $b \neq \bot$, else $\mathrm{pop}_i(\omega)$ is undefined,

$\mathrm{push}_{i,a}(\omega) = a[\omega_1]\omega$,

$\mathrm{change}_{i,a}(\omega) = a[\omega_1]\omega'$, if $b \neq \bot$ else $\mathrm{change}_{i,a}(\omega)$ is undefined.

For $k \geq 1$, $\mathcal{I}_k(\mathcal{A}_k) = \{\mathrm{pop}_i \mid i \in [1,k]\} \cup \{\mathrm{push}_{i,a}, \mathrm{change}_{i,a} \mid a \in A_i, i \in [1,k]\}$. is the set of instructions over $\mathcal{A}_k$.

Thus, given $\omega \in k\text{-}\mathsf{pds}$ and $i \leq k$, $\mathrm{pop}_i(\omega)$ erases $\mathrm{p}_i(\omega)$ on the top of the store, $\mathrm{push}_{i,a_i}(\omega)$ consists in add $a_i[\mathrm{p}_{i-1}(\omega)]$ on the top of the top i-pds and $\mathrm{change}_{i,a_i}(\omega)$ consists in replace $\mathrm{top}_i(\omega)$ by a_i.

Example 11. Let $\omega = b_3[b_2[b_1\ \bot]\ \bot_2]\ \bot_3$ be a 3-pds,
$\mathrm{pop}_3(\omega) = \bot_3$, $\mathrm{pop}_2(\omega) = b_3[\bot_2]\ \bot_3$, $\mathrm{pop}_1(\omega) = b_3[b_2[\bot]\ \bot_2]\ \bot_3$,
$\mathrm{push}_{2,a_2}(\omega) = b_3[a_2[b_1\ \bot]b_2[b_1\ \bot]\ \bot_2]\ \bot_3$,
$\mathrm{push}_{1,a_1}(\omega) = b_3[b_2[a_1b_1\ \bot]\ \bot_2]\ \bot_3$,
$\mathrm{change}_{3,a_3}(\omega) = a_3[b_2[b_1\ \bot]\ \bot_2]\ \bot_3$, $\mathrm{change}_{1,a_1}(\omega) = b_3[b_2[a_1\ \bot]\ \bot_2]\ \bot_3$.

We also define the inverse instruction of $\mathrm{push}_{i,a}$ which will be used to encode the k-pushdowns as words.

Definition 12. For any $i \geq 1$ and $a \in A_i$, the instruction of level i $\overline{\mathrm{push}}_{i,a}$ is defined for any $\omega \in i\text{-}\mathsf{pds}(\mathcal{A}_i)$ by

$\overline{\mathrm{push}}_{i,a}(\omega) = \omega'$ if there exists $\omega' \in i\text{-}\mathsf{pds}$ such that $\omega = \mathrm{push}_{i,a}(\omega')$

$\overline{\mathrm{push}}_{i,a}(\omega)$ is undefined else.

In other words, $\forall \omega \in k\text{-}\mathsf{pds}$,

$$\overline{\mathrm{push}}_{k,a}(\omega) = \omega' \text{ iff } \omega = a[\omega_1]b[\omega_1]\omega'' \text{ and } \omega' = b[\omega_1]\omega''.$$

1.4. Iterated pushdown machines

We define here *controlled iterated pushdowns systems* which extend systems with iterated storage structure intensively studied in the 70's (see [2, 22, 25, 26]) and more recently in [13, 15, 16, 24, 5, 7, 6, 18, 17, 19]. Here we define iterated pushdown machines whose transitions are conditioned by membership tests on the store.

Definition 13 (Controlled k-pushdown transitions system). Let $k \geq 0$, a k-TS is a structure $\mathfrak{A} = (Q, \mathcal{A}_k, \vec{C}, \Delta, q_0, F)$ where Q is a finite set of states, $\mathcal{A}_k$ is the sequence of pushdown alphabets, $\vec{C} = (C_1, \ldots, C_m)$ is a vector of *controllers* $C_i \subseteq k\text{-}\mathsf{pds}(\mathcal{A}_k)$, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is a set of final states and $\Delta \subseteq Q \times \mathrm{top}(k\text{-}\mathsf{pds}(\mathcal{A}_k)) \times \{0,1\}^m \times \mathcal{I}_k(\mathcal{A}_k) \times Q$ is a finite set of transitions.

The family of all k-TS controlled by $\vec{C}$ is $k\text{-}\mathsf{TS}^{\vec{C}}(A_1, \ldots, A_k)$. The set of *configurations* of $\mathfrak{A}$ is $Con_{\mathfrak{A}} = Q \times k\text{-}\mathsf{pds}(A_1, \ldots, A_k)$. The single step relation $\rightarrow_{\mathfrak{A}} \subseteq Con_{\mathfrak{A}} \times Con_{\mathfrak{A}}$ of $\mathfrak{A}$ is defined by

$$(p, \omega) \rightarrow_{\mathfrak{A}} (q, \omega') \text{ iff } (p, \mathrm{top}(\omega), \chi_{\vec{C}}(\omega), instr, q) \in \Delta, \text{ and } \omega' = instr(\omega).$$

We denote by $\stackrel{*}{\rightarrow}_{\mathfrak{A}}$ the reflexive and transitive closure of $\rightarrow_{\mathfrak{A}}$. The *set of k-pds generated by* $\mathfrak{A}$ is $\mathrm{P}(\mathfrak{A}) = \{\omega \in k\text{-}\mathsf{pds}(\mathcal{A}_k) |\ \exists q \in F,\ (q_0, \bot_k) \stackrel{*}{\rightarrow}_{\mathfrak{A}} (q, \omega)\}$.

1.5. Logics

1.5.1. Relational structures

Let $Sig = \{r_1, \ldots, r_n\}$ be a signature containing relational symbols only, where $\rho_i \in \mathbb{N}$ is the arity of symbol r_i, a (relational) **structure** $\mathcal{S}$ over the signature Sig consists of a domain $D_{\mathcal{S}}$ and relations $r_1, \ldots, r_n$ on $D_{\mathcal{S}}$ where ρ_i is the arity of r_i. We shall use three kind of structures:

Let $P \in A^*$ prefix closed and $t = \chi_P^{\vec{S}} \in P\text{-}\mathsf{Tree}(\{0,1\}^n)$,

- **Tree structures** Let $P \in A^*$ prefix closed and $t = \chi_P^{\vec{S}} \in P\text{-}\mathsf{Tree}(\{0,1\}^n)$, we associate to t the structure

  $$\underline{t} = \langle P, \varepsilon, (succ_a)_{a \in A}, S_1, \ldots, S_n \rangle,$$

  where $\forall a \in A$, $succ_a = \{(u, ua),\ u \in P,\ ua \in P\}$.

- **Image structures** Let t as previously and $f : P \rightarrow B$ be a map. We denote by $f(\underline{t})$ the relational structure

  $$f(\underline{t}) = \langle f(P), f(\varepsilon), (E_a)_{a \in A}, f(S_1), \ldots, f(S_n) \rangle,$$

  where $E_a = \{(f(u), f(ua)) \mid u, ua \in P\}$.

- **k-pds structure** Given $k \geq 1$, $\mathsf{PDS}_k(\mathcal{A}_k)$ is the structure whose domain is $k\text{-}\mathsf{pds}(\mathcal{A}_k)$ and endowed with the binary relations pop_i, $\mathrm{push}_{i,a}$ and $\mathrm{change}_{i,a}$ for every $1 \leq i \leq k$, $a \in A_i$. Relations pop_i, $\mathrm{push}_{i,a}$ and $\mathrm{change}_{i,a}$ are graphs of the corresponding instructions on pushdowns.

1.5.2. Monadic Second-Order Logic

Let Sig be a signature and $Var = \{x, y, z, \ldots, X, Y, Z \ldots\}$ be a set of variables, where $x, y, \ldots$ denote first-order variables and $X, Y, \ldots$ second-order variables. The set MSO(Sig) of MSO-formulas over Sig is the smallest set such that:

- $x \in X$ and $Y \subseteq X$ are MSO-formulas for every $x, Y, X \in Var$
- $r(x_1, \ldots x_\rho)$ is an MSO-formula for every $r \in Sig$, of arity ρ and every first order variables $x_1, \ldots x_\rho \in Var$
- if Φ, Ψ are MSO-formulas then $\neg\Phi$, $\Phi \vee \Psi$, $\exists x.\Phi$ and $\exists X.\Phi$ are MSO-formulas.

Let $\mathcal{S} = \langle D_\mathcal{S}, r_1, \ldots, r_n \rangle$ be a structure over the signature Sig, a valuation of Var over $D_\mathcal{S}$ is a function $val : Var \to D_\mathcal{S} \cup \mathcal{P}(D_\mathcal{S})$ such that for every $x, X \in Var$, $val(x) \in D_\mathcal{S}$ and $val(X) \subseteq D_\mathcal{S}$.
The satisfiability of an MSO-formula in the structure $\mathcal{S}$ with valuation val is then defined by induction on the structure of the formula, in the usual way.
An MSO-formula $\Phi(\bar{x}, \bar{X})$ (where $\bar{x} = (x_1, \ldots, x_\rho)$ and $\bar{X} = (X_1, \ldots, X_\tau)$ denotes free first and second-order variables of Φ) over Sig is said to be **satisfiable in** $\mathcal{S}$ if there exists a valuation val such that $\mathcal{S}, val \models \Phi(\bar{x}, \bar{X})$.
We will often abbreviate $\mathcal{S}, [\bar{x} \mapsto \bar{a}, \bar{X} \mapsto \bar{A}] \models \Phi(\bar{x}, \bar{X})$ by $\mathcal{S} \models \Phi(\bar{a}, \bar{A})$.

Definition 14. A structure $\mathcal{S}$ admits a decidable MSO-theory if for every MSO-sentence Φ (i.e. MSO-formula without free variables) one can effectively decide whether $\mathcal{S} \models \Phi$.

A vector $\vec{D} = (D_1, \ldots, D_m)$ of subsets of $D_\mathcal{S}$ is said to be **MSO-definable** in $\mathcal{S}$ iff there exists $\Phi(X_1, \ldots, X_m)$ in MSO(Sig) such that:

- $\mathcal{S} \models \Phi(D_1, \ldots, D_m)$ and
- $\forall \vec{S} = (S_1, \ldots, S_m)$, with $S_i \subseteq D_\mathcal{S}$, if $\mathcal{S} \models \Phi(S_1, \ldots, S_m)$ then $(S_1, \ldots, S_m) = (D_1, \ldots, D_m)$.

Remark that $\vec{D}$ is MSO-definable in $\mathcal{S}$ iff each D_i is MSO-definable in $\mathcal{S}$.

Definition 15. A structure $\mathcal{S}$ satisfies the property of **Definable Model** (or DM for short) if for every formula $\Phi(X_1, \ldots, X_n) \in$ MSO(Sig) satisfiable in $\mathcal{S}$, there exists $D_1, \ldots, D_n \subseteq D_\mathcal{S}$ such that

1. $\mathcal{S} \models \Phi(D_1, \ldots, D_n)$ and
2. $(D_1, \ldots, D_m)$ is MSO-definable in $\mathcal{S}$.

Let $Sig = \{r_1, \ldots, r_n\}$ (resp. $Sig' = \{r'_1, \ldots, r'_m\}$) be some relational signature and $\mathcal{S}$ (resp. $\mathcal{S}'$) be some structure over the signature Sig (resp. Sig').

Definition 16 (Interpretations). An MSO-*interpretation* of the structure $\mathcal{S}$ into the structure $\mathcal{S}'$ is an injective map $f : D_\mathcal{S} \to D_{\mathcal{S}'}$ such that,

1. $f(D_{\mathcal{S}})$ is MSO-definable in $\mathcal{S}'$
2. $\forall i \in [1,n]$, there exists $\Phi'_i(\bar{x}) \in MSO(Sig')$, (where $\bar{x} = x_1, \ldots, x_{\rho_i}$) fulfilling that, for every valuation val of Var in $D_{\mathcal{S}}$

$$(\mathcal{S}, val) \models r_i(\bar{x}) \Leftrightarrow (\mathcal{S}', f \circ val) \models \Phi'_i(\bar{x}).$$

Theorem 17 ([29]). *Suppose there exists a computable* MSO-**interpretation** *of the structure $\mathcal{S}$ into the structure $\mathcal{S}'$. If $\mathcal{S}'$ has a decidable* MSO*-theory, then $\mathcal{S}$ has a decidable* MSO*-theory too.*

Definition 18. If there exists a MSO-interpretation of $\mathcal{S}$ into $\mathcal{S}'$, and there exists a MSO-interpretation of $\mathcal{S}'$ into $\mathcal{S}$, then we say that $\mathcal{S}$ and $\mathcal{S}'$ are MSO-equivalent.

2. Monadic Second Order Logic and regular tree languages

2.1. Tree automata

We define here tree automata with p-oracle extending tree automata by allowing membership tests on nodes of input trees. As previously, for a given oracle $\vec{O}$, the application of any transition to a node u of a tree depends on the characteristic vector of u in $\vec{O}$.

Definition 19 (Tree automata with oracles). Let $m \geq 1$, a tree automaton with m oracles is a structure $\mathcal{A} = (Q, \Sigma, A, \vec{O}, \Delta, q_0, c)$ where Q, Σ and A are finite sets and $A = \{a_1, \ldots, a_n\}$, $\vec{O}$ is a vector of m subsets of A^*, $q_0 \in Q$, $c : Q \to [0, n_c]$, $n_c \geq 0$ and $\Delta \subseteq Q \times \Sigma \times \{0,1\}^m \times Q^n$.
Given $t \in A^*\text{-}\mathsf{Tree}(\Sigma)$, a *run* of $\mathcal{A}$ over t is a tree $r \in A^*\text{-}\mathsf{Tree}(Q)$ fulfilling:

$$r(\varepsilon) = q_0 \text{ and } \forall u \in A^*,\ (r(u), t(u), \chi^{\vec{O}}_{A^*}(u), r(ua_1), \ldots, r(ua_n)) \in \Delta.$$

A run r is successful if for every infinite path $\pi = q_1 \cdots q_n \cdots$ in r, the smallest $i \in [0, n_c]$ appearing infinitely often in the sequence $c(q_1), \ldots, c(q_n), \ldots$ is even. The tree language recognised by $\mathcal{A}$ is denoted $\mathrm{F}(\mathcal{A})$ and refers to the set of trees for which there exists a successful run.

The class of A^*-tree(Σ) automata with oracle $\vec{O}$ is denoted $\mathsf{TFA}^{\vec{O}}(A, \Sigma)$ (or $\mathsf{TFA}^{\vec{O}}(\Sigma)$ when A is understood), those of all $\vec{O}$-regular A^*-tree languages (i.e. recognised by automata in $\mathsf{TFA}^{\vec{O}}(A, \Sigma)$) is $\mathsf{TREG}^{\vec{O}}(A, \Sigma)$ (or $\mathsf{TREG}^{\vec{O}}(\Sigma)$ for short). Remark that a tree automaton with oracle $\emptyset$ is simply a tree automaton. We write then $\mathsf{TFA}(\Sigma)$ rather than $\mathsf{TFA}^{\emptyset}(\Sigma)$ and $\mathsf{TREG}(\Sigma)$ rather than $\mathsf{TREG}^{\emptyset}(\Sigma)$.

Definition 20 (Characteristic forest of $\vec{O}$). Given $\vec{O} = (O_1, \ldots, O_m)$, with $O_i \subseteq A^*$, the characteristic forest of $\vec{O}$ over Σ is $\mathrm{F}^{\vec{O}}_{\chi}(\Sigma) = A^*\text{-}\mathsf{Tree}(\Sigma)^\frown\{\chi^{\vec{O}}_{A^*}\}$.

Let us map each tree automaton with m oracles $\mathcal{A} = (Q, \Sigma, A, \vec{O}, \Delta, q_0, c) \in \mathsf{TFA}^{\vec{O}}$ to the tree automaton $\widetilde{\mathcal{A}} = (Q, \Sigma \times \{0,1\}^m, A, \widetilde{\Delta}, q_0, c) \in \mathsf{TFA}$ where $\widetilde{\Delta}$ consists of every transition $(q, (\alpha, \vec{o}), p_1, \ldots, p_n)$ such that $(q, \alpha, \vec{o}, p_1, \ldots, p_n) \in \Delta$. It can be easily checked that $\mathrm{F}(\mathcal{A}) = \pi_1(\mathrm{F}(\widetilde{\mathcal{A}}) \cap \mathrm{F}^{\vec{O}}_{\chi}(\Sigma))$.

Observation 21. For every $\vec{O} = (O_1, \ldots, O_m)$, $O_i \subseteq A^*$,

$$\mathsf{TREG}^{\vec{O}}(\Sigma) = \{\pi_1(F \cap \mathrm{F}^{\vec{O}}_{\chi}(\Sigma)) \mid F \in \mathsf{TREG}(\Sigma \times \{0,1\}^m)\}.$$

Remark 22. It can be easily seen that $\{\chi^{\vec{O}}_{A^*}\}$ and $\mathrm{F}^{\vec{O}}_{\chi}(\Sigma)$ are $\vec{O}$-regulars.

It is well known (see for example [28],[33]) that $\mathsf{TREG}(\Sigma)$ is closed under union, intersection, complementation and product. Then, we obtain from Observation 21:

Theorem 23. *The class* $\mathsf{TREG}^{\vec{O}}(A, \Sigma)$ *is closed under boolean operations and* $\mathsf{TREG}^{\vec{O}}(A, \Sigma_1 \times \Sigma_2)$ *is the class of all* $F_1 \widehat{} F_2$ *with* $F_1 \in \mathsf{TREG}^{\vec{O}}(A, \Sigma_1)$ *and* $F_2 \in \mathsf{TREG}^{\vec{O}}(A, \Sigma_2)$.

Given $P \subseteq A^*$ a prefix closed language, we define now automata recognising" P and for which the success of a given run depends only on nodes in P.

Definition 24 (P-cut automaton). An automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}$ is called P-cut if there exists a special state $q_\perp \in Q$ such that $c(q_\perp) = 0$ and
$\forall t \in A^*\text{-}\mathsf{Tree}(\Sigma)$, $r \in A^*\text{-}\mathsf{Tree}(Q)$ run of $\mathcal{A}$ over t, for every $u \in A^*$:

$$u \notin P \text{ iff } r(u) = q_\perp.$$

In this case, for every run r, subtrees external to P are successful and to know if r is successful, one only needs to test the parity condition on infinite paths inside P (then if P is finite, any run is always successful).

In the rest of the paper we use the notation $\mathsf{TREG}^{\vec{O}}_P(A, \Sigma)$ to refer to the class of forests $F_{|P}$, for $F \in \mathsf{TREG}^{\vec{O}}(A, \Sigma)$.

2.2. Tree languages as models of formulas

We adapt here the interpreted formalism of the MSO-logic of two successors (S2S) introduced in [28] to establish a correspondence between $\vec{O}$-regular forests and models of MSO-formulas over a labelled tree structure. For easier exposition, we shall restrict to binary trees (we denote $\mathsf{Tree}(\Sigma)$ instead of $\{0,1\}^*\text{-}\mathsf{Tree}(\Sigma)$). All definitions and results of this subsection can be naturally extended to the case where A is unspecified. In this subsection, $\vec{O}$ is always a vector $(O_1, \ldots, O_m)$, with $m \geq 1$ fixed and $O_i \subseteq \{0,1\}^*$ and P is a prefix closed subset of $\{0,1\}^*$.

We recall first the interpreted formalism of the MSO-logic of two successors by sticking to notations used in [33](Section 11).

Definition 25. An S2S-formula is an MSO-formula defined over the signature $(succ_0, succ_1)$, where $succ_i$ is a 2-ary relation symbol.
If $\phi(X_1, \ldots, X_m)$ is an S2S-formula and $t = \chi^{\vec{O}}_{\{0,1\}^*} \in \mathsf{Tree}_m$, write $\underline{t} \vdash \phi(X_1, \ldots, X_m)$ if $\langle \{0,1\}^*, succ_0, succ_1 \rangle, [X_i \mapsto O_i] \models \phi(X_1, \ldots, X_m)$.
Let $T(\phi) = \{t \in \mathsf{Tree}_m \mid \underline{t} \vdash \phi(X_1, \ldots, X_n)\}$. A tree language $F \in \mathsf{Tree}_m$ is called definable in S2S if $F = T(\phi)$ for some S2S-formula ϕ.

Theorem 26 ([28]). *The union of classes* $\mathsf{TREG}(\{0,1\}^n)$*, for* $n \geq 0$*, corresponds exactly to the class of tree languages definable in* S2S.

We now interpret S2S-formulas by fixing some free variables and interpreting formulas over restricted trees.

Definition 27. Let $\phi(X_1, \ldots, X_n)$ be an S2S-formula with $n > m$, we define the forest $T^{\vec{O}}_P(\phi)$ (or $T^{\vec{O}}$ if $P = \{0,1\}^*$) by:

$$T^{\vec{O}}_P(\phi) = \{t \in P\text{-}\mathsf{Tree}_{n-m} \mid \underline{t \widehat{\ } \chi^{\vec{O}}_P} \vdash \phi(X_1, \ldots, X_n)\}.$$

If $F = T^{\vec{O}}_P(\phi)$ for some S2S-formula ϕ, F is called definable in $\mathrm{S2S}^{\vec{O}}_P$ (or in $\mathrm{S2S}^{\vec{O}}$ if $P = \{0,1\}^*$).

Remark that $T^{\vec{O}}(\phi) = \pi_{1,n-m}(T(\phi) \cap F^{\vec{O}}_{\chi}(\{0,1\}^{n-m}))$. Then, using Observation 21, the Theorem 26 can easily be extended to the $\mathrm{S2S}^{\vec{O}}$ formalism.

Theorem 28. *The union of classes* $\mathsf{TREG}^{\vec{O}}(\{0,1\}^k)$*, for* $k \geq 1$*, corresponds exactly to the class of tree languages definable in* $\mathrm{S2S}^{\vec{O}}$.

Remark 29. Given ϕ an S2S-formula, the size (the number of states) of $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}$ such that $\mathrm{F}(\mathcal{A}) = T^{\vec{O}}(\phi)$ is the same as the size of $\widetilde{\mathcal{A}} \in \mathsf{TFA}$ such that $\mathrm{F}(\widetilde{\mathcal{A}}) = T(\phi)$. Then if ϕ has q quantifier alternations and its length is n, the size of $\mathcal{A}$ is $F(n,q)$, i.e., a tower $2^{2^{\cdot^{\cdot^{\cdot^{\mathcal{O}(n)}}}}}$ of height $q+1$ (see [20](§12.3)).

Observation 30. The structure $\underline{\chi^{\vec{O}}_{\{0,1\}^*}}$ has a decidable MSO-theory iff for all S2S-formula $\phi(X_1, \ldots, X_m)$, one can decide whether $\underline{\chi^{\vec{O}}_{\{0,1\}^*}} \vdash \phi(X_1, \ldots, X_m)$.

Corollary 31. *The emptiness problem is decidable for forests in* $\mathsf{TREG}^{\vec{O}}(\{0,1\}^k)$ *for all* $k \geq 1$ *iff* $\underline{\chi^{\vec{O}}_{\{0,1\}^*}}$ *has a decidable MSO-theory.*

Proof. $\underline{\chi^{\vec{O}}_{\{0,1\}^*}}$ has a decidable MSO-theory

iff for all S2S-formula $\phi(X_1, \ldots, X_m)$, one can decide whether $\underline{\chi^{\vec{O}}_{\{0,1\}^*}} \vdash \phi(X_1, \ldots, X_m)$

iff for all $k \geq 1$, for all S2S-formula $\psi(Y_1, \ldots, Y_m, X_1, \ldots, X_m)$, one can decide whether $\underline{\chi^{\vec{O}}_{\{0,1\}^*}} \vdash \exists Y_1, \ldots Y_k,\ \psi(Y_1, \ldots Y_k, X_1, \ldots, X_m)$

iff for all $k \geq 1$, for all S2S-formula $\psi(Y_1, \ldots, Y_m, X_1, \ldots, X_m)$, one can decide whether there exists $t \in \mathsf{Tree}_k$ such that $\underline{t\widehat{\ }\chi^{\vec{O}}_{\{0,1\}^*}} \vdash \psi(Y_1, \ldots Y_k, X_1, \ldots, X_m)$

iff for all $k \geq 1$, for all S2S-formula $\psi(Y_1, \ldots, Y_m, X_1, \ldots, X_m)$, one can decide whether $T^{\vec{O}}(\psi) = \emptyset$

iff for all $k \geq 1$, the emptiness problem is decidable for forests in $\mathsf{TREG}^{\vec{O}}(\{0,1\}^k)$.

We generalise now Theorem 28 to tree languages of domain P:

Theorem 32. *If P is* MSO*-definable in $\underline{\chi^{\vec{O}}_{\{0,1\}^*}}$, with $O_i \subseteq P$, then the union of classes $\mathsf{TREG}^{\vec{O}}_P(\{0,1\}^n)$ for $n \geq 1$ corresponds exactly to the class of* $\mathrm{S2S}^{\vec{O}}_P$*-definable tree languages.*

We start by proving the first implication:

Lemma 33. *If P is MSO-definable in $\underline{\chi^{\vec{O}}_{\{0,1\}^*}}$ then every tree language definable in $\mathrm{S2S}^{\vec{O}}_P$ belongs to $\mathsf{TREG}^{\vec{O}}_P(\{0,1\}^n)$ for some $n \geq 1$.*

Proof. Let $\phi(X_1, \ldots, X_n)$ be an S2S-formula, by relativizing ϕ to P, we construct an S2S-formula $\phi_P(X_1, \ldots, X_n)$ such that $\forall S_1, \ldots, S_n \subseteq \{0,1\}^*$,

$$\underline{\chi^{\vec{O}}_{\{0,1\}^*}} \models \phi_P(S_1, \ldots, S_n) \text{ iff } \underline{\chi^{\vec{O}}_P} \models \phi(S_1, \ldots, S_n) \text{ and } S_1, \ldots, S_n \subseteq P.$$

Let $F = T^{\vec{O}}_P(\phi)$ and $F' = T^{\vec{O}}(\phi_P)$, then $F = F'_{|P}$. From Theorem 28 applied to ϕ_P, the forest F' is $\vec{O}$-regular and thus $F \in \mathsf{TREG}^{\vec{O}}_P(\{0,1\}^n)$.

For the converse implication, we first restrict ourself to P-cut automata (see Definition 24).

Lemma 34. *For every P-cut automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}(\{0,1\}^n)$, the forest $\mathrm{F}(\mathcal{A})_{|P}$ is $\mathrm{S2S}^{\vec{O}}_P$-definable.*

Proof. Obviously, the proof of Theorem 28 also suits in the case of $\mathcal{A}$ is P-cut.

To achieve the proof of Theorem 32, it remains to show the following lemma.

Lemma 35. *Let $P \subseteq \{0,1\}^*$ prefix closed and $\vec{O}$ vector of subsets of P such that $P \in \mathsf{REG}^{\vec{O}}(\{0,1\})$. For every $F \in \mathsf{TREG}^{\vec{O}}(\Sigma)$, there effectively exists a P-cut automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}(\Sigma)$ such that $F_{|P} = \mathrm{F}(\mathcal{A})_{|P}$.*

PROOF. Let $P \cdot i^{-1} = \{u \in \{0,1\}^* | \, ui \in P\}$, for $i = 0, 1$ and $\vec{P} = (P \cdot 0^{-1}, P \cdot 1^{-1})$. Clearly, $\{\chi^{\vec{P}}_{\{0,1\}^*}\} \in \mathsf{TREG}^{\vec{O}}$. Theorem 23 ensures that there exists $\mathcal{A}_1 \in \mathsf{TFA}^{\vec{O}}(\Sigma \times \{0,1\}^2)$ such that $\mathrm{F}(\mathcal{A}_1) = F\widehat{\ }\{\chi^{\vec{P}}_{\{0,1\}^*}\}$. Then $\mathcal{A}_1$ allows to describe F, and also the *borders* of P. From $\mathcal{A}_1$, we construct now a new automaton $\mathcal{A}$ able to check from the borders of P, that there exists a labelling of subtrees external to P such that the complete tree belongs to F.

For every $q \in Q_1$ (the set of states of $\mathcal{A}_1$), we construct $\mathcal{A}_q \in \mathsf{TFA}$ whose transitions are all (p, α, q) such that $(p, \alpha, (0, \ldots, 0), q)$ is a transition of $\mathcal{A}_1$ and whose initial state is q. The emptiness problem being decidable for regular forests ([28],[33][Chapter 9]), we can effectively construct the set $Acc = \{q \in Q_1 \mid \mathrm{L}(\mathcal{A}_q) \neq \emptyset\}$.
Acc describes states from which, outside of P, one can find an accepting subtree. Then, $t \in F(\mathcal{A}_1)$ iff there exists a run r over t, such that

- every infinite path $r(u_1)r(u_2)\cdots r(u_n)\cdots$, with $u_i \in P$ is successful, and
- for all $u \in P$ such that $t(u) = (\alpha, b_0, b_1)$ and $b_i = 0$ for some $i = 1, 2$, $r(ui) \in Acc$.

Now, we construct an automaton $\mathcal{B}_1$ which is P-cut and such that $\mathrm{F}(\mathcal{A}_1)_{|P} = \mathrm{F}(\mathcal{B}_1)_{|P}$. We obtain this automaton by adding $q_\perp$ to the set of states (with $c(q_\perp) = 0$) and modifying the set of transitions of $\mathcal{A}_1$ in the following way: a transition $(p, (\alpha, b_0, b_1), \vec{o}, q_0, q_1)$ belongs to $Q_{\mathcal{B}}$ iff

- $b_0 = b_1 = 1$ and $(p, (\alpha, b_0, b_1), \vec{o}, q_0, q_1)$ belongs to $Q_{\mathcal{A}_1}$, or
- $\exists I \subseteq \{0,1\}$ such that $\forall i \in I$, $b_i = 0$ and $q_i = q_\perp$ and $\exists (p, (\alpha, b_0, b_1), \vec{o}, p_0, p_1) \in Q_{\mathcal{A}_1}$ such that for all i, $p_i = q_i$ if $i \notin I$ and $p_i \in Acc$ if $i \in I$.

From this new automaton, it is then easy to construct a P-cut automaton $\mathcal{A}$ recognising the language $\pi_1(\mathrm{F}(\mathcal{B}_1))$. We have then $\mathrm{F}(\mathcal{A})_{|P} = F_{|P}$. This conclude the proof of the Theorem 32.

Complexity analysis Suppose P is recognised by a word-automaton of size τ_P and F by a tree automaton of size τ. Then the size of $\mathcal{A}_1$ is $\tau \cdot \tau_P$, so is the P-cut automaton $\mathcal{A}$.
The construction of $\mathcal{A}$ requires to carry out $\tau\tau_P$ emptiness tests on automata of size smaller than that of $\tau \cdot \tau_P$. From [20][Cor 8.22], the emptiness test for a parity tree automaton of size s can be made in time $\mathcal{O}(|\Sigma| \cdot s^s)$. Then $\mathcal{A}$ can be constructed in time $\mathcal{O}(|\Sigma| \cdot (\tau \cdot \tau_P)^{\tau \cdot \tau_P})$.

Corollary 36. *The emptiness problem is decidable for forests in* $\mathsf{TREG}^{\vec{O}}_P(\{0,1\}^k)$, *for all* $k \geq 1$ *iff* $\underline{\chi^{\vec{O}}_P}$ *has a decidable MSO-theory.*

2.3. Regular trees and property of Definable Model

Regular trees form a remarkable family, as they correspond to unfolding of finite graphs, i.e., of graphs of finite automata. They are useful in several areas of computer science (see [10] for a survey on basic theory and applications

in semantics). We generalise here the notion of regular trees by defining trees corresponding to unfolding of graphs of p-oracle automata. We shall use these trees to study decidability and definability of the MSO-logic on labelled trees. We first extend regular trees to $\vec{O}$-**regular trees**. Each $\vec{O}$-regular tree corresponds to a deterministic word automaton with p-oracle $\vec{O}$. We then study links between existence of such a tree in a forest recognised with oracle-automata, and the satisfiability of the DM property for a labelled tree structures. Eventually, we close this subsection by defining tree automata *without input*. We show that the study of emptiness problem and regular trees can be restricted to such automata.

2.3.1. $\vec{O}$-Regular trees

A tree $t \in A^*\text{-}\mathsf{Tree}(\Sigma)$ is said to be $\vec{O}$-**regular** iff there exists a deterministic word automaton $\mathcal{A} \in \mathsf{FA}^{\vec{O}}(A)$ and a function $out : Q \to \Sigma$ *generating* t, i.e., such that $\forall u \in A^*$, $q \in Q$,

$$(q_{0,\uparrow}\, u) \to_{\mathcal{A}} (q, u_{\uparrow}) \text{ iff } out(q) = t(u).$$

Remark 37. The following remarks will be useful:

1. If $t \in A^*\text{-}\mathsf{Tree}(\Sigma)$ is $\vec{O}$-regular, then for every $\alpha \in \Sigma$, the set $L_\alpha = \{u |\ t(u) = \alpha\}$ is $\vec{O}$-regular.
2. For every $\vec{O}$, the characteristic tree $\chi^{\vec{O}}_{A^*}$ is $\vec{O}$-regular.

We extend this definition to P-trees by saying that any $t \in P\text{-}\mathsf{Tree}(\Sigma)$ is $\vec{O}$-regular when there exists $t' \in \mathsf{Tree}(\Sigma)$, $\vec{O}$-regular such that $t = t'_{|P}$.

2.3.2. Property of Definable Model

We study links between regular trees and the property of Definable Model (DM) formulated Definition 15.

Proposition 38. *If P is* MSO*-definable in $\underline{\chi^{\vec{O}}_{A^*}}$, the following properties are equivalent:*

1. $\underline{\chi^{\vec{O}}_{P}}$ *fulfils* DM*,*
2. *for all $n \geq 1$, for every non-empty forest $F \in \mathsf{TREG}^{\vec{O}}_P(\{0,1\}^n)$, there exists $\vec{D}$* MSO*-definable in $\underline{\chi^{\vec{O}}_{P}}$ such that F contains a $\vec{D}$-regular tree.*

Proof. Suppose P is MSO-definable in $\underline{\chi^{\vec{O}}_{A^*}}$, a simple rewriting of the DM property using Theorem 32 implies the equivalence between these two following properties:

(1) $\underline{\chi^{\vec{O}}_{P}}$ fulfils DM,

(2') for every non empty $F \subseteq \mathsf{TREG}_P^{\vec{O}}(\{0,1\}^n)$, there exists $\vec{S} = (S_1, \ldots, S_n)$ MSO-definable in $\underline{\chi_P^{\vec{O}}}$ such that the tree $\chi_P^{\vec{S}}$ belongs to F.

(2') $\Rightarrow$ **(2)** Suppose (2'), according to Remark 37(2), the tree $\chi_P^{\vec{S}}$ is $\vec{S}$-regular.

(2) $\Rightarrow$ **(2')** Suppose that $F \subseteq \mathsf{TREG}_{\vec{O}}^P$ is $\vec{O}$-regular and contains a $\vec{D}$-regular tree t. From definition of $\vec{D}$-regular tree, the language $\{t\}$ is $\vec{D}$-regular. Suppose that $t = \chi^{\vec{S}}{}_{|P}$, with $\vec{S} = (S_1, \ldots, S_n)$, Theorem 32 ensures that $\vec{S}$ is MSO-definable in $\underline{\chi_P^{\vec{D}}}$. Since $\vec{D}$ is MSO-definable in $\underline{\chi_P^{\vec{O}}}$, $\vec{S}$ is too.

Any nonempty regular forest contains a regular tree ([28], [33][Thm 9.3]), the following result is then a straightforward corollary of the proposition above.

Theorem 39. *For every finite alphabet A, the structure $\langle A^*, \varepsilon, (succ_a)_{a\in A}\rangle$ fulfills the property of Definable Model.*

2.3.3. Input-free tree automata

To deal with emptiness problems or existence of regular trees, one can without lost of generality work with **input-free tree automata** i.e., tree automata whose input alphabet is $\{\top\}$. In transitions of a such an automaton, the input letter can be omitted and then the set of transitions is $\Delta \subseteq Q \times \{0,1\}^m \times Q^{|A|}$. In the sequel, we write $\mathsf{TFA}^{\vec{O}}(A)$ (resp. $\mathsf{TFA}^{\vec{O}}$) rather than $\mathsf{TFA}^{\vec{O}}(A, \{\top\})$ (resp. $\mathsf{TFA}^{\vec{O}}(\{\top\})$).

Any tree automaton with m oracles $\mathcal{A} = (Q, \Sigma, A, \vec{O}, \Delta, q_0, c)$ can be transformed in $\mathcal{B} = (Q\times\Sigma, A, \vec{O}, \Delta', Q_0, c')$ input-free where for every $\alpha_1, \ldots, \alpha_n \in \Sigma$, $((q,\alpha), \vec{o}, (p_1,\alpha_1), \ldots (p_{|A|}, \alpha_{|A|})) \in \Delta'$ iff $(q, \alpha, \vec{o}, p_1, \ldots, p_{|A|}) \in \Delta$. Q_0 contains every (q_0, α), $\alpha \in \Sigma$ and $c'(q,\alpha) = c(q)$, $\forall \alpha \in \Sigma$. (It remains to reduce Q_0 to only one state, this construction being classical, we don't describe it). Obviously, successful runs of $\mathcal{B}$ are exactly pairs $r' = r\hat{}t$, where r is a successful run of $\mathcal{A}$ over t. We obtain then the following result which will permit to restrict next proofs to input-free automata:

Proposition 40. *For every $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}(A, \Sigma)$, one can find an input-free automaton $\mathcal{B} \in \mathsf{TFA}^{\vec{O}}(A)$ such that:*

1. $\mathrm{F}(\mathcal{A}) \neq \emptyset$ *iff there exists a successful run on $\mathcal{B}$,*
2. $\forall \vec{R}$, $\mathrm{F}(\mathcal{A})$ *contains a $\vec{R}$-regular tree iff there exists a successful $\vec{R}$-regular run on $\mathcal{B}$,*
3. $\forall P$ *prefix closed, if $\mathcal{A}$ is P-cut, then $\mathcal{B}$ is P-cut.*

Proposition 41. *For every $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}(A)$ input-free and deterministic, if there exists a run of $\mathcal{A}$ then this run is unique and $\vec{O}$-regular.*

Proof. Let us suppose that $\mathcal{A} = (Q, A, \vec{O}, \Delta, q_0, c)$, with $A = \{a_1, \ldots, a_n\}$, and consider the word-automaton $\mathcal{A}_r = (Q, A, \vec{O}, \Delta_r, q_0)$ where Δ_r consists of all transitions $(q, a_i, \vec{o}, p_i)$ such that $(q, \vec{o}, p_1, \ldots, p_n) \in \Delta$. Clearly, $\mathcal{A}_r$ is deterministic and if there exists a successful run of $\mathcal{A}$, it corresponds to the tree generated by $\mathcal{A}_r$ associated to the function $out : q \mapsto q$, $\forall q \in Q$.

3. Logic for restricted oracles

We now restrict our study to tree automata with oracles of the form $\vec{O} = (\mu_{|P}{}^{-1}(R_1), \ldots, \mu_{|P}{}^{-1}(R_m))$, where μ is a morphism from A^* to any semi-group S, P is a prefix closed subset of A^* and $R_i \subseteq S$.

We use a game-theoretical approach of these automata to express problems over $\vec{O}$-regular forests by means of MSO-formulas over the graph structure $\mu(\underline{\chi_P^{\vec{O}}})$ (see Section 1.5.1). This allows to show that if P is MSO-definable in $\underline{\chi^{\vec{O}}}$, then the MSO decidability can be transfered from $\mu(\underline{\chi_P^{\vec{O}}})$ to $\underline{\chi_P^{\vec{O}}}$. We also define a condition on $\mu_{|P}$ making possible the transfer of the property DM and such that the class of sets which are MSO-definable in $\underline{\chi_P^{\vec{O}}}$ corresponds exactly to class of $\mu_{|P}{}^{-1}(\vec{D})$-regular languages intersected with P, for any $\vec{D}$ MSO-definable in $\mu(\underline{\chi_P^{\vec{O}}})$.

3.1. Games for prefix-oracle automata

3.1.1. Parity game

A two-player game (player 0 and player 1) is a colored directed graph whose set of vertices V is partitioned in player 0's vertices (V_0) and player 1 ones (V_1), associated to a winning condition. Parity games are special games which have been much studied ([14, 20, 23])

Definition 42. A **parity game** is a tuple $G = (V_0, V_1, E, v_0, c)$ where $V = V_0 \biguplus V_1$ is the set of positions, $E \subseteq V \times V$ is the sets of possible moves, $v_0 \in V$ is the start position and $c : V \to [0, max]$ is a map associating to each vertex a priority by means of an integer which belongs to a bounded interval. A **play** in G is a (finite or infinite) path in the graph (V, E) starting at v_0. If the play is finite and ends in any vertex $v \in V_\epsilon$, $\epsilon \in \{0, 1\}$ (i.e., player ϵ cannot play anymore), then player ϵ is declared loser (and therefore the other player wins the play). Otherwise, the winner is determined by n_0, value of the minimal priority appearing infinitely often in the play. In other words, if the play is $v_0 v_1 \cdots v_n \cdots$, then n_0 is the smallest integer having an infinity of occurrences in the word $c(v_0)c(v_1)\cdots c(v_n)\cdots$. If n_0 is even, player 0 is declared winner, otherwise player 1 wins the play.

A **strategy** for player ϵ is a map $\mathrm{s} : (V^*V_\epsilon) \to V$ connecting any prefix of play $\rho = v_0 v_1 \ldots v_n$ to a vertex v_{n+1} such that $(v_n, v_{n+1}) \in E$. A strategy is **memoryless** if for any $\rho = v_0 v_1 \ldots v_n$ the value of $\mathrm{s}(\rho)$ depends only on the current vertex v_n. In this case, the strategy is represented as an application from V_ϵ to V. A play $\rho = v_0 v_1 \ldots v_n \ldots$ is said **conform** with s if for any $i \geq 0$ if $v_i \in V_\epsilon$, then $v_{i+1} = \mathrm{s}(v_0 \ldots v_i)$. A strategy s for player ϵ is a **winning strategy** if every play conform with s is won by player ϵ.

The notion of winning strategy allows to capture vertices from which a player is sure to win (if he chooses a good strategy). We say that player ϵ **wins the game** if there exists a **winning** strategy for ϵ

The following result will be useful in the sequel.

Theorem 43 ([14, 20]). *Given any parity game G:*

1. *one and only one player wins the game*
2. *for $\epsilon \in \{0,1\}$, if player ϵ wins the game, then player ϵ has a winning memoryless strategy.*

3.1.2. Games with p-oracle and regular trees

We use now parity games to express some problems related to $\vec{O}$-regular tree languages in the context fixed as follows:

- A is a finite alphabet, supposed reduced to two element: $A = \{a_0, a_1\}$ (all results established in this subsection remain true if A is unspecified),
- μ is a surjective morphism from A^* toward a semi-group $(M, \star)$,
- P is a prefix closed subset of A^*,
- $\vec{O} = ({\mu_{|P}}^{-1}(R_1), \ldots, {\mu_{|P}}^{-1}(R_m))$, with $m \geq 1$ and $R_i \subseteq M$.

We prove that the emptiness problem for $\vec{O}$-regular tree languages reduced to determine the winner of a parity game whose vertices are included in the product of $\mu(P)$ with a finite set. From this result, we show (Proposition 53(1)) that the emptiness problem for $\vec{O}$-regular tree languages reduced to the satisfiability of an MSO-formula in $\mu(\underline{\chi_P^{\vec{O}}})$ (see Section 1.5.1). We prove, in addition, that every non-empty $\vec{O}$-regular tree language contains a $\mu_{|P}(\vec{D})$-regular tree, where $\vec{D}$ is MSO-definable in $\mu(\underline{\chi_P^{\vec{O}}})$ (Proposition 53(2)).

We first restrict ourself to the study of input-free P-cut automata. Advantage of using P-cut automata is that to know if there exists a run and if there exists a successful run, there is only need to consider nodes in P. In addition, in a run of a P-cut automaton, nodes which do not belong to P are indicated by the label $q_\perp$.

Definition 44 (Game with p-oracles). Given an input-free P-cut automaton $\mathcal{A} = (Q, A, \vec{O}, \Delta, q_0, c)$, we construct the parity game $G_{\mathcal{A}} = (V_0, V_1, E, v_0, c')$ where
$V_0 = \mu(P) \times Q$ and $V_1 = \mu(P) \times \Delta$, $v_0 = (\mu(\varepsilon), q_0)$
$E = E_0 \cup E_1$ where $E_0 \subseteq V_0 \times V_1$, $E_1 \subseteq V_1 \times V_0$ and
$E_0 = \{((m,p),(m,\delta)) \mid \delta = (p, \chi_{\mu(\vec{O})}(m), p_0, p_1) \in \Delta\}$,
$E_1 = \{((m,\delta),(m \star \mu(a_i), p_i)) \mid p_i \neq q_\perp,\ \delta = (p, \vec{o}, p_0, p_1),\ i \in \{0,1\}\}$ and
c' is defined by $c'(m,p) = c(p)$ and $c'(m,(p,\vec{o},p_0,p_1)) = c(p)$.

Each player moves alternately in the game. In position $(\mu(u), p)$, player 0 chooses a transition $\delta = (p, \vec{o}, p_0, p_1)$ from those fulfilling $\vec{o} = \chi_{A^*}^{\vec{O}}(u)$. He moves then to $(\mu(u), \delta)$. Now it's the player 1 turn to play, he chooses a direction a_i to follow ($i \in \{0,1\}$) and moves to $(\mu(ua_i), p_i)$. Hence, μ being a morphism, for every prefix of play ending in $(m,x) \in V$, then $m = \mu(u)$ where u consists of the sequence of directions chosen by player 1.

Lemma 45. *For any input-free P-cut automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}(A)$, $\mathcal{A}$ has a successful run iff player 0 has a winning strategy in $G_{\mathcal{A}}$.*

Proof. Let r be a run on $\mathcal{A}$, and s_r the strategy defined by: $\forall v_0 \dots v_n$ prefix of a play with $v_n \in V_0$ and such that the sequence of directions chosen by player 1 is $u \in A^*$,

$$\mathrm{s}_r(v_0 \dots v_n) = (\mu(u), (r(u), \chi_{\vec{O}}(u), r(ua_0), r(ua_1))).$$

Clearly, s_r is winning iff r is successful.

Conversely, given any winning strategy s for player 0 we construct the tree r_{s} by applying to each vertex u, the transition given by $\mathrm{s}(\rho_u)$ where ρ_u is the prefix of play conform with s whose sequence of selected directions is u:

- $r_{\mathrm{s}}(\varepsilon) = q_0$ and $\rho_\varepsilon = (\mu(\varepsilon), q_0)$,
- $\forall u \in P$, if $\mathrm{s}(\rho_u) = (\mu(u), (p, \vec{o}, p_0, p_1))$, then $\forall i \in \{0, 1\}$, $r_{\mathrm{s}}(ua_i) = p_i$ and $\rho_{ua_i} = \rho_u \cdot \mathrm{s}(\rho_u) \cdot (\mu(u) \star \mu(a_i), p_i)$,
- $\forall u \notin P$, $r_{\mathrm{s}}(u) = q_\perp$.

The tree thus constructed is obviously a run on $\mathcal{A}$ and is successful iff s is a winning strategy.

Thus by applying Theorem 43:

Lemma 46. *For any input-free P-cut automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}(A)$, $\mathcal{A}$ has a successful run iff player 0 has a memoryless winning strategy in $G_{\mathcal{A}}$.*

Given a memoryless strategy s for player 0, we consider the game $G^{\mathrm{s}}_{\mathcal{A}} = (\emptyset, V_0 \cup V_1, E^{\mathrm{s}}, c')$, where $E^{\mathrm{s}} = E_1 \cup \{(v, v') \mid v \in V_0, \mathrm{s}(v) = v'\}$. Without loss of generality, we can suppose that $\mathcal{A}$ is *complete*, i.e., for every $(q, \vec{o})$, there exists a transition $(q, \vec{o}, q_1, q_2)$. In this case, any finite play in $G_{\mathcal{A}}$ ends in a player 1's position, and is then winning for player 0.

Lemma 47. *For every complete input-free P-cut automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}$, any memoryless strategy for player 0 s is winning in $G_{\mathcal{A}}$ iff the reduced game $G^{\mathrm{s}}_{\mathcal{A}}$ is winning for player 0.*

Proof. An infinite sequence of vertices $v_0 \cdots v_n \cdots$ is a play in $G^{\mathrm{s}}_{\mathcal{A}}$ iff it is a play in $G_{\mathcal{A}}$ conforms with s. Since vertices of $G_{\mathcal{A}}$ and $G^{\mathrm{s}}_{\mathcal{A}}$ have same priority, an infinite sequence of vertices is a winning play in $G_{\mathcal{A}}$ conform with s iff it is a winning play in $G^{\mathrm{s}}_{\mathcal{A}}$.

Any parity game $G = (V_0, V_1, E, v_0, c)$ with $c : V \to [0, max]$ is naturally associated to a relational structure $\boldsymbol{G}$ of domain V defined over the signature $\mathcal{G} = (V_0, V_1, E, v_0, c_0, \dots, c_{max})$, where for all $i \in [0, max]$, the arity of c_i is 1.

Lemma 48. *For every complete input-free P-cut automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}$, one can find an MSO($\mathcal{G}_\mathcal{A}$)-sentence* WIN *such that* $\forall$S *memoryless strategy for player 0 in $G_\mathcal{A}$,*

$$\mathbf{G}^{\mathrm{S}}_\mathcal{A} \models \text{WIN} \textit{ iff } \mathrm{S} \textit{ is winning.}$$

PROOF. We construct a formula P_0 such that $\mathbf{G}^{\mathrm{S}}_\mathcal{A} \models P_0$ iff there exists a play in $G^{\mathrm{S}}_\mathcal{A}$ lost by player 0, i.e., iff there exists an infinite path $v_0 \cdots v_n \cdots$ such that the smallest integer appearing infinitely often in $c'(v_0) \cdots c'(v_n) \cdots$ is odd.

$P_0 := \exists X, X_0, \ldots, X_{max}$,

1. X is a path containing v_0
2. $\forall n, X_n = \{x \text{ appearing infinitely often in } X \mid c_n(x)\}$
3. the smallest n such that $X_n \neq \emptyset$ is odd

Using [20][§12.2], "being a path" is MSO-expressible in $G^{\mathrm{S}}_\mathcal{A}$. From Theorem 43, player 0 loses the game iff player 1 wins the game, hence WIN $:= \neg P_0$.

We relate now these results to the MSO-logic of the structure $\mu(\underline{\chi^{\vec{O}}_P})$ by means of an encoding the subsets of V_1, and an encoding of the player 0's memoryless strategies, with a vector of subsets of $\mu(P)$. Given $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}$, we fix the following notations:

- $\Delta = \{\delta_1, \ldots, \delta_d\}$, where $\delta_i \neq \delta_j$ for all $i \neq j$,
- $Q = \{s_1, \ldots, s_\tau\}$,
- for every $D \subseteq V$, $g(D) = (g_1(D), \ldots, g_d(D), h_1(D), \ldots, h_\tau(D))$, where $\forall i \in [1, \ldots d]$, $j \in [1, \tau]$,

$$g_i(D) = \{\sigma \mid (\sigma, \delta_i) \in D\},\ h_j(D) = \{\sigma \mid (\sigma, s_j) \in D\},$$

- we associate to any player 0's memoryless strategy S the vector $\vec{S}_{\mathrm{S}} = (S_{\mathrm{S},1}, \ldots, S_{\mathrm{S},d})$, where $\forall i \in [1, d]$,

$$S_{\mathrm{S},i} = \{\sigma \in \mu(P),\ \mathrm{S}(\sigma, \pi_1(\delta_i)) = (\sigma, \delta_i)\},$$

  and denote $\vec{\mathcal{S}}_{\mathrm{S}}$ the vector $\mu^{-1}_{|P}(\vec{S}_{\mathrm{S}})$.

Remark that $\vec{S}_{\mathrm{S}}$ gives a complete characterization of S.

Lemma 49. *Given an input-free P-cut automaton $\mathcal{A} \in \mathsf{TFA}^{\vec{O}}$,* S *a memoryless strategy for player 0 in $G_\mathcal{A}$, and $\phi(X_1, \ldots, X_n)$ an MSO($\mathcal{G}_\mathcal{A}$)-formula, one can effectively construct an MSO-formula ϕ^g such that $\forall D_1, \ldots D_n \subseteq \mu(P) \times (Q \cup \Delta)$*

$$\mathbf{G}^{\mathrm{S}}_\mathcal{A} \models \phi(D_1, \ldots, D_n) \textit{ iff } \mu(\underline{\chi^{\vec{\mathcal{S}}_{\mathrm{S}}}_P}) \models \phi^g(g(D_1), \ldots, g(D_n)).$$

PROOF. Let us construct ϕ^g if ϕ is an atomic formula:

- $\forall i \in [1, d]$, $j \in [1, \tau]$, σ, $\sigma' \in \mu(P)$,

- $E^{\mathrm{s}}((\sigma,\delta_i),(\sigma',p_j))$ iff $\exists\epsilon\in\{0,1\}$ s.t. $\sigma'=\sigma\star\mu(a_\epsilon)$ and $\pi_{2+\epsilon}(\delta_i)=p_j$
- $E^{\mathrm{s}}((\sigma,p_j),(\sigma',\delta_i))$ iff $\sigma'=\sigma$, $\pi_1(\delta_i)=p_j$ and $\sigma\in S_{\mathrm{s},i}$

Then $(E^{\mathrm{s}})^g(X_1,\dots,X_{p+\tau},Y_1,\dots,Y_{p+\tau})$ can be expressed in the following way: $\exists i\in[1,d]$, $\exists j\in[1,\tau]$ such that

- either $X_i=\{x\}$, $Y_{d+j}=\{y\}$ and the other ones are empty and $\exists\epsilon\in\{0,1\}$ s.t. $y=x\star\mu(a_\epsilon)$ and $\pi_{2+\epsilon}(\delta_i)=p_j$
- or $Y_i=\{y\}$, $X_{d+j}=\{x\}$ and the other ones are empty and $y=x$, $\pi_1(\delta_i)=p_j$ and $x\in S_{\mathrm{s},i}$

- $c^g_{\mathrm{s},n}(X_1,\dots,X_{p+\tau})$ corresponds to the XOR of the two following properties:
  - $\exists i\in[1,d]$ such that $X_i=\{x\}$ and the other ones are empty and $c(\delta_i)=n$,
  - $\exists j\in[1,\tau]$ such that $X_{d+j}=\{x\}$ and the other ones are empty and $c(p_j)=n$.
- if $\phi(X,Y):=X\subseteq Y$,
  then $\phi^g(X_1,\dots,X_{d+\tau},Y_1,\dots,Y_{d+\tau}):=\forall i\in[1,d+\tau],X_i\subseteq Y_i$.

Finally, if ϕ is not atomic, ϕ^g is given by an obvious induction.

Combining Lemma 48 and Lemma 49, we obtain:

Lemma 50. *Let $\mathcal{A}\in\mathsf{TFA}^{\vec{O}}$ complete, input-free and P-cut, one can find an MSO-sentence* SG*, such that for every memoryless strategy* S *for player 0 in $G_{\mathcal{A}}$,*

$$\mu(\underline{\chi_P^{\vec{S}_{\mathrm{s}}}})\models \mathrm{SG} \textit{ iff } \mathrm{S} \textit{ is winning.}$$

Given $\vec{D}=(D_1,\dots,D_d)$ any vector of subsets of $\mu(P)$, it is easy to determine if there exists a memoryless strategy S such that $\vec{D}$ encodes S (i.e such that $\vec{D}=\vec{S}_{\mathrm{s}}$). Indeed, if we suppose $\mathcal{A}$ is complete, there is just to check that for every $\sigma\in\mu(P)$, for every state p, there exists one and only one $i\in[1,d]$ such that, the first component of δ_i is p, and $\chi^{\mu(\vec{O})}(\sigma)=\pi_2(\delta_i)$ and $\sigma\in D_i$. This property can be easily expressed in MSOL, hence, we deduce from Lemma 50:

Lemma 51. *For every complete P-cut input-free automaton $\mathcal{A}\in\mathsf{TFA}^{\vec{O}}$, there exists $d\geq 0$ and an* MSO*-formula* $\mathrm{REG}_{\mathcal{A}}(X_1,\dots,X_d)$ *such that $\forall\vec{S}=(S_1,\dots,S_d)$, $S_i\subseteq\mu(P)$, the following properties are equivalent:*

1. $\mu(\underline{\chi_P^{\vec{O}}})\models\mathrm{REG}_{\mathcal{A}}(S_1,\dots,S_d)$
2. *$\vec{S}$ encodes a winning memoryless strategy for player 0 in $G_{\mathcal{A}}$.*

Let us associate to $\mathcal{A}$ and any memoryless strategy S, a **deterministic** input-free tree automaton $\mathcal{A}_{\mathrm{s}}=(Q,A,\vec{\mathcal{S}}_{\mathrm{s}},\Delta_{\mathrm{s}},q_0)$, where Δ_{s} is constructed in the following way:

- $\forall \vec{o}$, $(q_\perp, \vec{o}, q_\perp, q_\perp) \in \Delta_{\mathrm{s}}$
- if $\delta_i = (q, \vec{o}, p_0, p_1) \in \Delta$, then $(q, \vec{b}, p_0, p_1) \in \Delta_{\mathrm{s}}$, $\forall \vec{b} \in \{0,1\}^d$ such that
  - $b_i = 1$,
  - $\forall j \neq i$, $b_j = 0$ if the first component of δ_j is q.

This automaton follows the transitions of $\mathcal{A}$ indicated by the strategy S.

Lemma 52. *For every winning memoryless strategy* S *for player 0, $\mathcal{A}_{\mathrm{s}}$ is deterministic, P-cut and its unique run is a successful run of $\mathcal{A}$.*

Proof. By choice of test vectors, $\mathcal{A}_{\mathrm{s}}$ is clearly deterministic. $\mathcal{A}$ being complete and from definition of $\vec{S}_{\mathrm{s}}$, $\mathcal{A}_{\mathrm{s}}$ admits a run r since $\forall u \in P$, there exists $i \in [1,d]$ such that $r(u) = \delta_i$ and $\mu(u) \in S_{\mathrm{s},i}$. We prove that r is a successful run of $\mathcal{A}$:

- $r(\varepsilon) = q_0$,
- $\forall u \in A^*$, the transition $(r(u), \chi_{A^*}^{\vec{S}_{\mathrm{s}}}(u), r(ua_0), r(ua_1))$ belongs to Δ_{s} and there exists then $i \in [1,d]$ and $\vec{o}$ such that $\delta_i = (r(u), \vec{o}, r(ua_0), r(ua_0)) \in \Delta$ and $S_{\mathrm{s},i} = 1$. Hence, we have in addition $\mathrm{s}(\mu(u), r(u)) = (\mu(u), \delta_i)$ and then $\vec{o} = \chi^{\vec{O}}(u)$.

Then, r is a run of $\mathcal{A}$, and since S is winning, r is successful and $\mathcal{A}_{\mathrm{s}}$ is P-cut.

Since $\mathcal{A}_{\mathrm{s}}$ is deterministic, Proposition 41 implies that its unique run is a $\vec{S}_{\mathrm{s}}$-regular tree. Applying Lemma 40, these results can be extended to the case of automata with inputs. In addition, from Lemma 35, if P is MSO-definable in $\underline{\chi_{A^*}^{\vec{O}}}$, for every $\mathcal{B} \in \mathsf{TFA}^{\vec{O}}(A)$, there exists a P-cut automaton $\mathcal{A}$ in $\mathsf{TFA}^{\vec{O}}(A)$ such that $\mathrm{F}(\mathcal{A})_{|P} = \mathrm{F}(\mathcal{B})_{|P}$. Hence, when P is MSO-definable in $\underline{\chi_{A^*}^{\vec{O}}}$, Lemma 52 and Lemma 51 can be extended to every automaton in $\mathsf{TFA}^{\vec{O}}$. The following proposition summarize these results.

Proposition 53. *For every forest $F \in \mathsf{TREG}_P^{\vec{O}}(A, \Sigma)$, where P is MSO-definable in $\underline{\chi_{A^*}^{\vec{O}}}$, there exists $d \geq 0$ and a formula $\mathrm{REG}_F(X_1, \ldots, X_d)$, such that*

1. *$F \neq \emptyset$ iff $\mu(\underline{\chi_P^{\vec{O}}}) \models \exists X_1, \ldots, X_d \cdot \mathrm{REG}_F(X_1, \ldots, X_d)$*
2. *for every $\vec{S} = (S_1, \ldots, S_d)$, $S_i \subseteq \mu(P)$,*
   if $\mu(\underline{\chi_P^{\vec{O}}}) \models \mathrm{REG}_F(S_1, \ldots, S_d)$, then F contains a $\mu_{|P}{}^{-1}(\vec{S})$-regular tree.

Complexity analysis.
Lemma 51: Let $\mathcal{A}$ be an automaton fulfilling the conditions of Lemma 51, and suppose τ is the number of states of $\mathcal{A}$. The formula $\mathrm{REG}_{\mathcal{A}}$ defined Lemma 51 contains 3 quantifier changes and its length is $\mathcal{O}(\tau)$. Indeed, using [20](§12.2), the formula WIN constructed in Lemma 48 contains 3 quantifier alternations and its length is constant. The formula SG obtained in Lemma 50 has then 3 quantifier changes and has length $\mathcal{O}(1)$. Finally, the transformation of SG in

REG does not modify the number of quantifier changes but adds $\mathcal{O}(\tau)$ symbols. Then REG has 3 quantifier changes and has length $\mathcal{O}(\tau)$.

Proposition 53: Suppose that τ is the number of states of an automaton recognizing F and τ' is the number of states of the word-automaton recognizing P. Using Lemma 35 we construct in time $\mathcal{O}(|\Sigma| \cdot (\tau \cdot \tau')^{\tau \cdot \tau'})$ a P-cut automaton $\mathcal{A}$ having $\tau \cdot \tau'$ states and such that $\mathrm{F}(\mathcal{A})_P = F_P$. This automaton can be transformed in an input-free automaton having $|\Sigma|^{|A|} \cdot \tau \cdot \tau'$ states.
Finally, using the complexity analysis of Lemma 51, the formula REG_F defined in Proposition 53 contains 3 quantifier changes and its length is $\mathcal{O}(|\Sigma|^{|A|} \cdot \tau \cdot \tau')$. This formula can be constructed in time $\mathcal{O}(|\Sigma| \cdot (\tau \cdot \tau')^{\tau \cdot \tau'})$.

3.2. Transfer theorems

We use the Proposition 53 to transfer some properties of the structure $\mu(\chi_P^{\vec{O}})$ toward the structure $\underline{\chi_P^{\vec{O}}}$. The following definition fixes hypothesis for which these results hold.

Definition 54 (Transfer Hypothesis (TH)). We write $\mathrm{TH}(\mu_{|P}, \vec{O})$ if:

$\mu : A^* \to M$ is a surjective morphism of semi-group, P is a prefix closed language in A^*, and there exists $\vec{R}$ vector of subsets of $\mu(P)$ such that $P \in \mathsf{REG}^{{\mu_{|P}}^{-1}(\vec{R})}(A)$ and $\vec{O} = \mu_{|P}^{-1}(\vec{R})$.

Theorem 55 (Transfer of decidability). *Let μ be a morphism from A^* to any semi-group, $P \subseteq A^*$ be a prefix closed language, and $\vec{O}$ a vector of subsets of P, such that* $\mathrm{TH}(\mu_{|P}, \vec{O})$.
If the MSO*-theory of $\mu(\underline{\chi_P^{\vec{O}}})$ is decidable, then the* MSO*-theory of $\underline{\chi_P^{\vec{O}}}$ is decidable.*

Proof. Suppose that the MSO-theory of $\mu(\underline{\chi_P^{\vec{O}}})$ is decidable. For all $F \in \mathsf{TREG}_P^{\vec{O}}$, one can decide whether $\mu(\underline{\chi_P^{\vec{O}}}) \models \exists \vec{X}, \mathrm{REG}_F(\vec{X})$ where REG_F is the formula established in Proposition 53, i.e., whether F is empty. Hence, from Corollary 31, the MSO-theory of $\underline{\chi_P^{\vec{O}}}$ is decidable.

Complexity analysis.
Let $\mathcal{C}_D(n, \tau)$ be the time to decide the validity of a sentence in $\mathrm{MSO}(\underline{\chi_P^{\vec{O}}})$ of length n and having τ quantifier alternations. Suppose that P is recognized by an automaton having τ_P states. Let ϕ be a sentence in $\mathrm{MSO}(\underline{\chi_{|P}^{\vec{O}}})$ of length n and having τ quantifier changes. From Remark 29, we can compute a tree automaton $\mathcal{A} \in \mathsf{TREG}^{\vec{O}}$ such that $T(\phi) = T(\mathcal{A})$, and having $F(n, \tau)$ states. Then, the formula $\mathrm{REG}_{T(\mathcal{A})}$ has 3 quantifier alternations and length $|\Sigma|^{|A|} F(n, \tau) \tau_P$ and is constructed in time $\mathcal{O}(|\Sigma|(F(n, \tau)\tau_P)^{F(n, \tau)\sigma_P})$. Finally, we decide if ϕ is true in time $\mathcal{C}_D(3, |\Sigma|^{|A|} F(n, \tau)\tau_P) + \mathcal{O}(|\Sigma|(F(n, \tau)\tau_P)^{F(n,q)\tau_P})$ (or $\mathcal{C}_D(3, F(n, \tau))$ if $P = A^*$).

We define now a condition on $\mu_{|P}$ allowing to transfer the DM property (see Definition 15).

Definition 56. Given any surjectif morphism μ from A^* into any semi-group, and $P \subseteq A^*$ prefix closed, the restricted map $\mu_{|P}$ is said to be MSO-**invertible** if for every $\vec{O}$ vector of subsets of P, for every $D \subseteq \mu(P)$,
if D is MSO-definable in $\mu(\underline{\chi_P^{\vec{O}}})$ then ${\mu_{|P}}^{-1}(D)$ is MSO-definable in $\underline{\chi_P^{\vec{O}}}$.

Theorem 57 (Transfer of property DM). *If* TH$(\mu_{|P}, \vec{O})$ *and* $\mu_{|P}$ *is* MSO-*invertible, then* $\mu(\underline{\chi_P^{\vec{O}}})$ *satisfies* DM *implies* $\underline{\chi_P^{\vec{O}}}$ *satisfies* DM.

Proof. Let F be a non-empty $\vec{O}$-regular forest in P-Tree_n, from Proposition 53 and since $\mu(\underline{\chi_P^{\vec{O}}})$ fulfills DM, there exists $\vec{S}$ such that F contains a ${\mu_{|P}}^{-1}(\vec{S})$-regular tree and $\vec{S}$ is MSO-definable in $\mu(\underline{\chi_P^{\vec{O}}})$. Since $\mu_{|P}$ is MSO-invertible, there exists $\vec{D} = {\mu_{|P}}^{-1}(\vec{S})$ such that $\vec{D}$ is MSO-definable in $\underline{\chi_P^{\vec{O}}}$ and F contains a $\vec{D}$-regular tree. Hence, from Proposition 38, $\underline{\chi_P^{\vec{O}}}$ fulfills DM.

Complexity analysis For a formula in MSO$(\underline{\chi_P^{\vec{O}}})$ of length n and having τ quantifier alternations, we denote by $\mathcal{C}_M(n,\tau)$ the time needed to construct a formula that defines a model, and by $(n_{\mathcal{S}}, \tau_{\mathcal{S}})$ the size of a formula that defines inverse models. Suppose P is recognized by an automaton having τ_P states. Given $\phi \in$ MSO$(\underline{\chi_P^{\vec{O}}})$ of length n and having τ quantifier changes, one can construct a formula that defines a model of ϕ in time $\mathcal{C}_M(3, F(n,\tau)) + \mathcal{O}((2F(n,\tau)\cdot\tau_P)^{F(n,\tau)\cdot\tau_P})$ (or $\mathcal{C}_M(3, F(n,\tau))$ if $P = A^*$).

Theorem 58 (Structure Theorem). *If* $\mu_{|P}$ *is* MSO-*invertible,* TH$(\mu_{|P}, \vec{O})$ *and* $\mu(\underline{\chi_P^{\vec{O}}})$ *satisfies* DM*, then for every* $L \subseteq P$*, the following properties are equivalent:*

- L *is* MSO-*definable in* $\underline{\chi_P^{\vec{O}}}$
- *there exists* $\vec{D}$ MSO-*definable in* $\mu(\underline{\chi_P^{\vec{O}}})$ *such that* L *is* ${\mu_P}^{-1}(\vec{D})$-*regular.*

Proof. Let us suppose that L is MSO-defined in $\underline{\chi_P^{\vec{O}}}$ by a formula $\phi(X)$. Then from Theorem 32, there exists a $\vec{O}$-regular forest $F = \{\chi_P^L\}$ such that $F = T_P^{\vec{O}}(\phi)$. From Proposition 53 and since $\mu(\underline{\chi_P^{\vec{O}}})$ fulfills DM, there exists $\vec{D}$ such that χ_P^L is ${\mu_{|P}}^{-1}(\vec{D})$-regular and $\vec{D}$ is MSO-definable in $\mu(\underline{\chi_P^{\vec{O}}})$. From Remark 37.1, the language L is ${\mu_{|P}}^{-1}(\vec{D})$-regular.

Conversely, given $\vec{D}$, MSO-definable in $\mu(\underline{\chi_P^{\vec{O}}})$, then since ${\mu_{|P}}^{-1}$ is MSO-invertible, ${\mu_{|P}}^{-1}(\vec{D})$ is MSO-definable in $\underline{\chi_P^{\vec{O}}}$. Given L a ${\mu_{|P}}^{-1}(\vec{D})$-regular language, by using the automata-characterization of L, it is then easy to find a MSO-formula defining L in $\underline{\chi_P^{\vec{O}}}$.

Complexity analysis.
For a formula in $\mathrm{MSO}(\underline{\chi_P^{\vec{O}}})$ of length n and having τ quantifier alternations, we denote by $\mathcal{C}_M(n,\tau)$ the time needed to construct a formula that defines a model, and by $(n_{\mathcal{S}},\tau_{\mathcal{S}})$ the size of a formula that defines inverse models. Suppose that P is recognized by an automaton having τ_P states and that L is defined by $\phi \in \mathrm{MSO}(\underline{\chi_P^{\vec{O}}})$ of length n and having τ quantifier changes, one can construct a word oracle-automaton recognizing L having $F(3,F(n,\tau))$ states. It can be constructed in time $\mathcal{C}_M(3,F(n,\tau)) + \mathcal{O}((2F(n,\tau)\cdot\tau_P)^{F(n,\tau)\cdot\tau_P})$ (or $\mathcal{C}_M(3,F(n,\tau))$ if $P = A^*$).

3.3. A first example of application

Various authors have exhibited classes of relation $R \subseteq \mathbb{N}$ for which $\langle \mathbb{N}, 0, +1, R\rangle$ has a decidable MSO-theory. Cite for recent examples [8, 19, 17]. These structures can be seen as images by a morphism, in order to transfer the decidability toward a tree structure. Given two alphabets A and B, with $B \subseteq A$, we consider the map $l_B : A^* \to \mathbb{N}$ associating to every word in A^* its number of occurrences of letters in B. This map is clearly a surjective morphism when $\mathbb{N}$ is endowed with the "+" operator.

Corollary 59. *For every $\vec{N} = (N_1,\ldots,N_m)$, $N_i \subseteq \mathbb{N}$ such that $\langle \mathbb{N}, +1, \vec{N}\rangle$ has a decidable* MSO*-theory, the structure $\underline{\chi_{A^*}^{{l_B}^{-1}(\vec{N})}}$ admits a decidable* MSO*-theory.*

This result has already been proved in [34][Proposition 2] for the case $A = B$, as a direct application of the results about unfolding of graphs obtained in [11, 12]. However, to our knowledge, this method does not allow the transfer of the DM property, nor to deal with the decidability for the case $A \neq B$. Conversly, Theorem 55 does not seems to cover all results we can obtain by unfolding, since the unfolded graph must induce a structure of semi-group isomorphic to a tree.

Corollary 60. *Given $\vec{N}$ a vector of subsets of $\mathbb{N}$,*

1. *the structure $\underline{\chi_{A^*}^{l_B^{-1}(\vec{N})}}$ fulfills* DM*,*
2. *sets MSO-definable in $\underline{\chi_{A^*}^{l_B^{-1}(\vec{N})}}$ are languages in $\mathsf{REG}^{l_B^{-1}(\vec{D})}$, for $\vec{D}$ MSO-definable in $\langle \mathbb{N}, 0, +1, \vec{N}\rangle$.*

Proof. We prove these results by using Theorems 57 and 58.
Since $l_B(\underline{\chi_{A^*}^{l_B^{-1}(\vec{N})}}) = \langle \mathbb{N}, 0, +1, \vec{N}\rangle$, we just need to prove that

1. $\langle \mathbb{N}, 0, +1, \vec{N}\rangle$ fulfills the DM property:
   This result is proved for all $\vec{N}$ in [30].
2. l_B is MSO-invertible:
   Consider the covering $\mathcal{R}$ of A^* consisting of all sets R that form an infinite path from ε and containing an infinite number of elements in B. For all $R \in \mathcal{R}$, the restriction of l_B to R is a surjective map from R to $\mathbb{N}$. Remark

in addition that the property "be an element of $\mathcal{R}$" can be expressed by an MSO-formula over $\langle A^*, \varepsilon, (\mathrm{SUCC}_a)_{a\in A}\rangle$.
For every MSO-formula ϕ over $\mathcal{S} = \langle \mathbb{N}, 0, +1, \vec{N}\rangle$ with n free variables, we construct by induction an MSO-formula ϕ' over $\mathcal{S} = \langle \mathbb{N}, 0, +1, \vec{N}\rangle$ with $n+1$ free-variables satisfying for every $R \in \mathcal{R}$, for every $S_1, \ldots, S_n \subseteq A^*$:

$$\underline{\chi_{A^*}^{l_B^{-1}(\vec{N})}} \models \phi'(S_1 \cap R, \ldots, S_n \cap R, R) \text{ iff } \mathcal{S} \models \phi(l(S_1 \cap R, \ldots, S_n \cap R)) \quad (1)$$

In this proof, relations with free first order variables are replaced by "equivalent" relations with free second order variables. For exemple, $\varepsilon(x)$ is replaced by $\varepsilon(X) == \exists x.\ X = \{x\} \wedge \varepsilon(x)$.

- $0'(X, Y) := \varepsilon(X)$,
- $(+1)'(X_1, X_2, Y) := \bigvee_{b\in B} \mathrm{SUCC}_b(X_1, X_2)$,
- if $\varphi := \exists X, \psi(X, X_1, \ldots, X_n)$, then $\varphi' := \exists X, \psi'(X \cap Y, X_1, \ldots, X_n, Y)$,
- if $\varphi := \forall X, \psi(X, X_1, \ldots, X_n)$, then $\varphi' := \exists X, \psi'(X \cap Y, X_1, \ldots, X_n, Y)$,

cases $X_1 \subseteq X_2$ and $X_1 \subseteq N$ are given in an obvious way, idem for boolean combinations: $(\phi \vee \psi)' := \phi' \vee \psi'$, $(\phi \wedge \psi)' := \phi' \wedge \psi'$ et $(\neg\phi)' := \neg\phi'$. It is easy to check that atomic formulas fulfill Equivalence (1). We treat only the case $\varphi = \exists X \cdot \psi$. For the universal quantifier, the proof is similar. For boolean combination, the proof is obvious
Fix $S_1, \ldots, S_n \subseteq A^*$ and $R \in \mathcal{R}$.

$$\begin{aligned}
\underline{\chi_{A^*}^{l_B^{-1}(\vec{N})}} &\models \varphi'(S_1 \cap R, \ldots, S_n \cap R, R) \text{ iff} \\
\exists S \subseteq A^*,\ \underline{\chi_{A^*}^{l_B^{-1}(\vec{N})}} &\models \psi'(S \cap R, S_1 \cap R, \ldots, S_n \cap R, R) \text{ iff (by (i.h))} \\
\exists S \subseteq A^*,\ \mathcal{S} &\models \psi(l_B(S \cap R), l_B(S_1 \cap R), \ldots, l_B(S_n \cap R)) \text{ iff} \\
\exists D \subseteq \mathbb{N},\ \mathcal{S} &\models \psi(D, l_B(S_1 \cap R), \ldots, l_B(S_n \cap R)) \text{ iff} \\
\mathcal{S} &\models \exists X,\ \psi(X, l_B(S_1 \cap R), \ldots, l_B(S_n \cap R)) \text{ iff} \\
\mathcal{S} &\models \varphi(l_B(S_1 \cap R), \ldots, l_B(S_n \cap R)).
\end{aligned}$$

We have remarked $\mathcal{R}$ is a covering of A^* and for all $R \in \mathcal{R}$, l_B restricted to R is surjective, then $\forall D \subseteq A^*$, $D' \subseteq \mathbb{N}$,

$$D = l_B^{-1}(D') \text{ iff } \forall R \in \mathcal{R},\ l_B(R \cap D) = D'.$$

Then, for every formula $\phi(X_1, \ldots, X_n)$ over $\mathcal{S}$, the formula

$$\phi_{l_B}(X_1, \ldots, X_n) := \forall R \in \mathcal{R},\ \phi'(R \cap X_1, \ldots, R \cap X_n, R)$$

fulfills $\underline{\chi_{A^*}^{l_B^{-1}(\vec{N})}} \models \phi_{l_B}(D_1, \ldots, D_n)$ iff $\exists D'_1, \ldots, D'_n$ such that $\mathcal{S} \models \phi(D'_1, \ldots, D'_n)$ and for every $i \in [1, n]$, $l_B^{-1}(D_i) = D'_i$.

4. Words, iterated-pushdowns and tree-structures

In order to apply results obtained above to iterated pushdowns, we need to represent k-pds as words in a prefix closed language. We then encode each $\omega \in k\text{-}\mathsf{pds}(A_1, \ldots, A_k)$ by a word representing the smallest instructions sequence of $\text{push}_{i,a}$ and $\overline{\text{push}}_{i,a}$ computing ω from $\bot_k$. The set of such encodings is a prefix closed language over $\widehat{A_{1,k}}$ denoted $\mathcal{P}_k$.

We use transfer theorems proved in the previous section to study MSO-properties of the structure $\boldsymbol{\mathcal{P}_k} = \langle \mathcal{P}_k, \varepsilon, (\bullet_a)_{a \in \widehat{A_{1,k}}} \rangle$ where $\bullet_a$ is the binary relation *right-product* by a inside the free group $(\text{Irr}(A_{1,k}), \bullet, \varepsilon)$. We show that for every $k \geq 1$, the structure $\boldsymbol{\mathcal{P}_k}$ has a decidable MSO-theory and fulfills the DM property (Theorem 76). We also define a class of automata with p-oracles recognizing exactly sets which are MSO-definable inside $\boldsymbol{\mathcal{P}_k}$ (Theorem 79). Eventually we prove that the structure PDS_k is MSO-equivalent to the structure $\boldsymbol{\mathcal{P}_k}$. It follows that PDS_k has a decidable MSO-theory and fulfills DM. We also give a definition of **regular sets of k-pushdowns** which enjoy several nice characterizations (Theorem 85).

4.1. Iterated-pushdowns viewed as words

Let $A_1, \ldots, A_k, \ldots$ be store alphabets and $A_0 = \emptyset$, $\forall k \geq 0$, we denote by $A_{1,k}$ the union of $A_1, \ldots, A_k$. Every $\omega \in k\text{-}\mathsf{pds}(A_1, \ldots, A_k)$ can be represented by a word on $\widehat{A_{1,k}} = \overline{A_{1,k}} \cup A_{1,k}$ encoding an instructions sequence computing ω from $\bot_k$:

- every $a \in A_i$ corresponds to $\text{push}_{i,a}$
- every $\bar{a} \in \overline{A}_i$ corresponds to $\overline{\text{push}}_{i,a}$

For instance, the 2-pds $\omega = a_2[c_1 b_1 \ \bot] a_2[a_1 \ \bot] \ \bot \ [\bot]$ can be represented by the word $u_1 = a_2 a_1 a_2 \bar{a}_1 b_1 c_1$, or by $u_2 = a_2 a_1 b_2 \bar{b}_2 a_2 \bar{a}_1 b_1 c_1$, or by $u_3 = a_2 a_1 a_1 b_2 \bar{b}_2 \bar{a}_1 a_2 \bar{a}_1 b_1 c_1$.
There are then several representations of the same k-pds but all have the same reduced representative in $(\text{Irr}(A_{1,k}), \bullet, \varepsilon)$. Each k-pds will be encoded by its reduced representation. In the previous example, the reduced representation is u_1 (since $\rho(u_1) = \rho(u_2) = \rho(u_3) = u_1$). Each word in $\widehat{A_{1,k}}^*$ does not define a *valid* sequence of instructions. For example, $a_1 b_1 a_2 \bar{b}_1 \bar{b}_1$ is not valid since $a_1 b_1 a_2 \bar{b}_1$ correspond to $a_2[a_1 \ \bot] \ \bot \ [b_1 a_1 \ \bot]$ and $\overline{\text{push}}_{b_1,1}$ is then undefined.
Let us introduce the set $\mathcal{M}_k$ of words in $\widehat{A_{1,k}}^*$ encoding all valid sequences of moves, as well as the set $\mathcal{P}_k$ of reduced words of $\mathcal{M}_k$ which encodes the set of k-pds. We define simultaneously $\mathcal{P}_k(A_1, \ldots, A_k)$ (or simply $\mathcal{P}_k$ when the A_i's are fixed) and $\mathcal{M}_k(A_1, \ldots A_k)$ (or simply $\mathcal{M}_k$) by induction on k:

- $\mathcal{P}_0 = \{\varepsilon\}$,
- $\forall k \geq 0$, $\mathcal{M}_k(A_1, \ldots, A_k) = \{u \in \widehat{A_{1,k}}^* \mid \forall v \preccurlyeq u, \rho(v) \in \mathcal{P}_k(A_1, \ldots, A_k)\}$ and
  $\mathcal{P}_{k+1}(A_1, \ldots, A_{k+1}) = \{u \in \rho((\widehat{A_{1,k}} \cup A_{k+1})^*) \mid \pi_{\widehat{A_{1,k}}}(u) \in \mathcal{M}_k(A_1, \ldots, A_k)\}$.

Clearly, $\mathcal{P}_1(A_1) = A_1^*$.

Definition 61 (Projection). For $k \geq 0$, $\mathrm{f}_k : \mathcal{P}_{k+1} \to \mathcal{P}_k$ is defined for every $u \in \mathcal{P}_{k+1}$ by $\mathrm{f}_k(u) = \rho(\pi_{\widehat{A_{1,k}}}(u))$. We extend f_k by $\mathrm{f}_{i+1,k} : \mathcal{P}_{i+1} \to \mathcal{P}_k$ obtained by successive applications of $\mathrm{f}_i, \mathrm{f}_{i-1}, \ldots, \mathrm{f}_k$.

An obvious induction on k proves the following recursive definition of $\mathcal{P}_k$

Proposition 62. *For every $k \geq 1$, $u \in \mathcal{P}_k$ and $a \in A_i$, $1 \leq i \leq k$,*

$u \bullet a \in \mathcal{P}_k$ *and*

$u \bullet \bar{a} \in \mathcal{P}_k$ *iff* $\mathrm{f}_{k,i}(u) \in \mathcal{P}_i \cdot a$.

For every $k \geq 0$, sets $\mathcal{P}_k$ and k-pds are linked by a bijection denoted φ_k:

Definition 63. The map $\varphi_k : k\text{-pds}(A_1, \ldots, A_k) \to \mathcal{P}_k(A_1, \ldots, A_k)$ is defined by induction on $k \geq 0$ by:

- $\varphi_0(\bot_0) = \varepsilon$,
- $\forall k \geq 0$, $\omega_1 \in k$-pds, $\omega \in (k+1)$-pds and $a \in A_{k+1}$,
  - $\varphi_{k+1}(\bot\ [\omega_1]) = \varphi_k(\omega_1)$
  - $\varphi_{k+1}(a[\omega_1]\omega) = (\varphi_{k+1}(\omega) \cdot a \cdot \overline{\mathrm{f}_k(\varphi_{k+1}(\omega))}) \bullet \varphi_k(\omega_1)$.

Example 64. Let ω_{ex} be the following 3-pds:
$\omega_{ex} = a_3[b_2[b_1a_1\ \bot]a_2[a_1\ \underline{\bot]\ \bot_2]\ \bot\ [a_2}[a_1\ \bot]a_2[\bot]\ \bot_2] = a_3[\omega_1]\omega$
Then, $\varphi_3(\omega_{ex}) = (\varphi_3(\omega)a_3\overline{f_2(\varphi_3(\omega))}) \bullet \varphi_2(\omega_1)$.
We have, $\varphi_2(b_2[b_1a_1\ \bot]a_2[a_1\ \bot]\ \bot_2) = a_2a_1b_2b_1$ and $\varphi_2(a_2[a_1\ \bot]a_2[\bot]\ \bot_2\ ) = a_2a_2a_1$, then $\varphi_3(\omega) = a_2a_2a_1$. We obtain then,

$$\begin{aligned}\varphi_3(\omega_{ex}) &= a_2a_2a_1a_3\overline{a_2a_2a_1} \bullet (a_2a_1b_2b_1)\\ &= a_2a_2a_1a_3(\bar{a_1}\,\bar{a_2}\,\bar{a_2}) \bullet (a_2a_1b_2b_1)\\ &= a_2a_2a_1a_3\bar{a_1}\,\bar{a_2}a_1b_2b_1.\end{aligned}$$

Proposition 65. *For every $(k+1)$-pds $\omega = a[\omega_1]\omega'$, $\varphi_k(\omega_1) = \mathrm{f}_k(\varphi_{k+1}(\omega))$.*

Proof. From definition of φ_k and f_k:

$$\begin{aligned}\mathrm{f}_k(\varphi_{k+1}(a[\omega_1]\omega')) &= \mathrm{f}_k(\varphi_{k+1}(\omega') \cdot a \cdot \overline{\mathrm{f}_k(\varphi_{k+1}(\omega'))} \bullet \mathrm{f}_k(\varphi_k(\omega_1)))\\ &= \mathrm{f}_k(\varphi_{k+1}(\omega')) \bullet \overline{\mathrm{f}_k(\varphi_{k+1}(\omega'))} \bullet \varphi_k(\omega_1)\\ &= \varphi_k(\omega_1).\end{aligned}$$

Remark 66. From Proposition 65 and definition of φ_{k+1}, it appears clearly that for all $\omega = a_n[\omega_n]\cdots a_1[\omega_1]\ \bot\ [\omega_0] \in (k+1)$-pds, $n \geq 0$,
$\varphi_{k+1}(\omega) = \varphi_k(\omega_0)a_1\overline{\varphi_k(\omega_0)} \bullet \varphi_k(\omega_1)a_2\overline{\varphi_k(\omega_1)} \bullet \varphi_k(\omega_2)\cdots a_n\overline{\varphi_k(\omega_{n-1})} \bullet \varphi_k(\omega_n)$.

Lemma 67. *For every $k \geq 0$, φ_k is a bijective map.*

PROOF. **Injection:** Let us sketch by induction on $k \geq 0$ that φ_k is an injective map. If $k = 0$, it is obvious. Suppose φ_k injective, for $k \geq 0$. For every $\omega, \omega' \in (k+1)$-pds having same image by φ_{k+1}, Remark 66 implies the following decompositions:
$\omega = a_n[\omega_n] \cdots a_1[\omega_1] \perp [\omega_0]$ et $\omega' = a_n[\omega'_n] \cdots a_1[\omega'_1] \perp [\omega'_0]$, $n \geq 0$.
We check φ_{k+1} is bijective by a second induction over $n \geq 0$. If $n = 0$, the induction hypothesis over k proves that $\omega = \omega'$. Else, let $\omega = a_n[\omega_n]\omega''$ and $\omega' = a_n[\omega_n]\omega'''$. From definition of φ_{k+1} and by hypothesis $\varphi_{k+1}(\omega) = \varphi_{k+1}(\omega')$:
$\varphi_{k+1}(\omega'') \cdot a \cdot \overline{\mathrm{f}_k(\varphi_{k+1}(\omega''))} \bullet \mathrm{f}_k(\varphi_k(\omega_n)) = \varphi_{k+1}(\omega''') \cdot a \cdot \overline{\mathrm{f}_k(\varphi_{k+1}(\omega'''))} \bullet \mathrm{f}_k(\varphi_k(\omega'_n))$,
in other words,
$\varphi_{k+1}(\omega'') = \varphi_{k+1}(\omega''')$ et $\overline{\mathrm{f}_k(\varphi_k(\omega''))} \bullet \mathrm{f}_k(\varphi_k(\omega_n)) = \overline{\mathrm{f}_k(\varphi_k(\omega'''))} \bullet \mathrm{f}_k(\varphi_k(\omega'_n))$.
From induction hypothesis over n, we obtain $\omega'' = \omega'''$ and since $(\mathrm{Irr}(A_{1,k}), \bullet, \varepsilon)$ is a group, $\mathrm{f}_k(\varphi_k(\omega_n)) = \mathrm{f}_k(\varphi_k(\omega'_n))$. Then $\omega = \omega'$.
Surjection:let us define inductively the map ${\varphi_k}^{-1}$:

- ${\varphi_0}^{-1}(\varepsilon) = \perp_0$
- for every $u \in \mathcal{P}_{k+1}$, $k \geq 0$,
  - if $u \in \mathcal{P}_k$, then ${\varphi_{k+1}}^{-1}(u) = \perp [{\varphi_k}^{-1}(u)]$
  - else there exists $u' \in \mathcal{P}_{k+1}$, $a \in A_{k+1}$, $u_1 \in \mathrm{Irr}(A_{1,k})$ such that $u = u'au_1$ and

$$ {\varphi_{k+1}}^{-1}(u) = a[{\varphi_k}^{-1}(f_k(u))]{\varphi_{k+1}}^{-1}(u') $$

We check that ${\varphi_k}^{-1}$ really defines the inverse map of φ_k by induction over k. We detail the case $u = u'au_1$:

$$ \begin{array}{ll} & \varphi_{k+1}({\varphi_{k+1}}^{-1}(u)) \\ = & \varphi_{k+1}(a[{\varphi_k}^{-1}(\mathrm{f}_k(u))]\varphi_{k+1}^{-1}(u')) \\ = & \varphi_{k+1}({\varphi_{k+1}}^{-1}(u')) \cdot a \cdot \overline{\mathrm{f}_k(\varphi_{k+1}({\varphi_{k+1}}^{-1}(u')))} \bullet \varphi_k(\varphi_k^{-1}(\mathrm{f}_k(u))) \\ = & u' \cdot a \cdot \overline{\mathrm{f}_k(u')} \bullet \mathrm{f}_k(u) = u' \cdot a \cdot \overline{\mathrm{f}_k(u')} \bullet \mathrm{f}_k(u') \bullet u_1 = u' \cdot a \cdot u_1. \end{array} $$

We close this section by studying links between the right-product in $\mathcal{P}_k$ and the application of instructions to k-pushdowns.

Lemma 68. *For every $k \geq 1$, $u, v \in \mathcal{P}_k$, and $a \in A_i$, $1 \leq i \leq k$,*

$$ v = u \bullet a \text{ \textit{iff} } {\varphi_k}^{-1}(v) = \mathrm{push}_{i,a}({\varphi_k}^{-1}(u)) $$

$$ v = u \bullet \bar{a} \text{ \textit{iff} } {\varphi_k}^{-1}(u) = \mathrm{push}_{i,a}({\varphi_k}^{-1}(v)) \text{ \textit{iff} } {\varphi_k}^{-1}(v) = \overline{\mathrm{push}}_{i,a}({\varphi_k}^{-1}(u)) $$

4.2. Logic on a free group

Let $A_1, \ldots, A_k, \ldots$ be disjoints alphabets fixed for the rest of the paper and Sig_k the signature $(\varepsilon, (\bullet_a)_{a \in \widehat{A_{1,k}}})$ where ε and $\bullet_a$ are respectively a unary and a binary relation. The signature Sig_k^m is Sig_k augmented with m unary relations. Consider the structure $\boldsymbol{\mathcal{P}_k}$ defined on Sig_k whose domain is $\mathcal{P}_k(A_1, \ldots, A_k)$ and such that $\forall a \in \widehat{A_{1,k}}$, $\bullet_a = \{(u, v) |\ u, v \in \mathcal{P}_k, v = u \bullet a\}$. For every $\vec{O} = (O_1, \ldots, O_m)$ with $O_i \subseteq \mathcal{P}_k$, $\boldsymbol{\mathcal{P}_k}^{\vec{O}}$ denotes the structure $\boldsymbol{\mathcal{P}_k}$ augmented with relations $O_1, \ldots, O_m$.

By using the fact that, $\forall k \geq 1$, $\boldsymbol{\mathcal{P}_k}$ is the image by f_k of the tree of domain $\mathcal{P}_{k+1}$, we show inductively, by applying transfert theorems of Section 3, that $\boldsymbol{\mathcal{P}_k}$ satisfies the property DM, that its MSO-theory is decidable and we give an automata-characterization of the MSO-definable sets of $\boldsymbol{\mathcal{P}_k}$.

4.2.1. MSO-*invertibility*

It is proved here that for every $k \geq 1$, the mapping f_k is MSO-invertible (see Definition 56). This result will be helpful in two ways: first to apply transfer theorems to $\boldsymbol{\mathcal{P}_k}$ and latter to show that structures $\boldsymbol{\mathcal{P}_k}$ and PDS_k are MSO-equivalent.

In the sequel, we denote by T_k the structure $\langle \mathcal{P}_k, \varepsilon, (succ_a)_{a \in \widehat{A_{1,k} \cup \overline{A_{k-1}}}} \rangle$. In addition, $\forall \vec{O}$ vector of subsets of $\mathcal{P}_k$, we write $T_k^{\vec{O}}$ the structure obtained by adding to T_k the unary relations O_i.

Observation 69. $\forall k \geq 1$, $\forall \vec{O}$ vector of subsets of $\mathcal{P}_{k+1}$, $\mathrm{f}_k(T_{k+1}^{\vec{O}}) = \boldsymbol{\mathcal{P}_k}^{\mathrm{f}_k(\vec{O})_1}$.

We proceed in a similar way as the proof of Corollary 60. We start by defining a partition of $\mathcal{P}_{k+1}$ whose each element is in bijection with $\mathcal{P}_k$. For every $k \geq 1$, we consider the family $\mathfrak{F}_{k+1}$ consisting of all sets $F \subseteq \mathcal{P}_{k+1}$ such that:

- either $F = \mathcal{P}_k$,
- or $\exists u \in \mathcal{P}_{k+1}$ and $\exists a \in A_{k+1}$ such that $F = \{uaw \in \mathcal{P}_{k+1} |\ w \in \mathrm{Irr}(A_{1,k})\}$.

Each $F \in \mathfrak{F}_{k+1}$ encodes, via φ_{k+1}, a maximal set of $(k+1)$-pds which differ only by top level-k elements. The family $\mathfrak{F}_{k+1}$ allows to recompose the inverse image of any definable subset of $\mathcal{P}_k$. The proof is based on the following remarks:

Remark 70. For every $k \geq 1$:

1. $\mathfrak{F}_{k+1}$ defines a partition of $\mathcal{P}_{k+1}$.
2. For every $F \in \mathfrak{F}_{k+1}$, the restriction of f_k to F is a bijection toward $\mathcal{P}_k$.

[1] Here, and in the rest of the paper, f_k is extended to sets, and vector of sets in a natural way.

3. The property $F \in \mathfrak{F}_{k+1}$ is MSO-expressible in $\boldsymbol{\mathcal{P}_{k+1}}$.
   Indeed, $F \in \mathfrak{F}_{k+1}$ iff F is a maximal set such that for every $u, v \in F$, there exists a path from u to v using only edges $\bullet_a$ where $a \in \widehat{A_{1,k}}$. It is then easy to construct a MSO-formula whose set of models in $\boldsymbol{\mathcal{P}_{k+1}}$ is exactly $\mathfrak{F}_{k+1}$.

Lemma 71. *Given $k \geq 1$ and $m \geq 0$, for every* MSO*-formula $\phi(X_1, \ldots, X_n)$ over Sig_k^m, there exists a* MSO*-formula $\phi'(X_1, \ldots, X_n, Y)$ over Sig_{k+1}^m such that*
$\forall \vec{R} = (R_1, \ldots, R_m), R_i \subseteq \mathcal{P}_k$*:*
$\forall S_1, \ldots, S_n \in \mathcal{P}_{k+1}$, $\forall F \in \mathfrak{F}_{k+1}$,

$$\boldsymbol{\mathcal{P}_{k+1}}^{\mathrm{f}_k{}^{-1}(\vec{R})} \models \phi'(S_1 \cap F, \ldots, S_n \cap F, F) \text{ iff } \boldsymbol{\mathcal{P}_k}^{\vec{R}} \models \phi(\mathrm{f}_k(S_1 \cap F, \ldots, S_n \cap F)).$$

Proof. We construct ϕ' for ϕ atomic, the other cases are given by the same induction as the proof of Corollary 60.

- $(\varepsilon)'(X, Y) := \exists x \mid X = \{x\} \wedge \bigwedge_{a \in A_{1,k}} \neg(\exists y,\, x \bullet \bar{a} = y)$,

- $\forall a \in \widehat{A_{1,k}}$, $(\bullet_a)'(X_1, X_2, Y) := \bullet_a(X_1, X_2)$.
  Let $a \in \widehat{A_{1,1,k}}$. If $F = \mathcal{P}_k$, clearly $\forall u, v \in \mathcal{P}_k$

  $$\boldsymbol{\mathcal{P}_{k+1}}^{\mathrm{f}_k{}^{-1}(\vec{R})} \models \bullet_a{}'(\{u\}, \{v\}, F) \text{ iff } \boldsymbol{\mathcal{P}_k}^{\vec{R}} \models \bullet_a(\{\mathrm{f}_k(u)\}, \{\mathrm{f}_k(v)\}).$$

  If $F = \{ubw \in \mathcal{P}_{k+1} \mid w \in \mathrm{Irr}(A_k)\}$, with $b \in A_{k+1}$, then for every $v = ubw$, $v' = ubw' \in F$,

  $$\begin{array}{lll} \boldsymbol{\mathcal{P}_{k+1}}^{\mathrm{f}_k{}^{-1}(\vec{R})} \models (\bullet_a)'(\{v\}, \{v\}', F) & \text{iff} & \boldsymbol{\mathcal{P}_{k+1}}^{\mathrm{f}_k{}^{-1}(\vec{R})} \models \bullet_a(\{ubw\}, \{ubw'\}) \\ & \text{iff} & w' = w \bullet a \\ & \text{iff} & \mathrm{f}_k(ub) \bullet w' = \mathrm{f}_k(ub) \bullet w \bullet a \\ & \text{iff} & \boldsymbol{\mathcal{P}_k}^{\vec{R}} \models \bullet_a(\{\mathrm{f}_k(v)\}, \{\mathrm{f}_k(v')\}). \end{array}$$

- if $\phi(X) := X \subseteq R_i$, then $\phi'(X) := X \subseteq \mathrm{f}_k{}^{-1}(R_i)$. Indeed, $\forall S \subseteq \mathcal{P}_{k+1}$,

  $$\begin{array}{lll} \boldsymbol{\mathcal{P}_{k+1}}^{\mathrm{f}_k{}^{-1}(\vec{R})} \models \phi'(S \cap F, F) & \text{iff} & \boldsymbol{\mathcal{P}_{k+1}}^{\mathrm{f}_k{}^{-1}(\vec{R})} \models S \cap F \subseteq \mathrm{f}_k{}^{-1}(R_1) \\ & \text{iff} & \mathrm{f}_k(S \cap F) \subseteq R_1 \\ & \text{iff} & \boldsymbol{\mathcal{P}_k}^{\vec{R}} \models \phi(\mathrm{f}_k(S)). \end{array}$$

- if $\Phi(X_1, X_2) := X_1 \subseteq X_2$, then $\Phi'(X_1, X_2, Y) := X_1 \subseteq X_2$. The proof is the same as the previous case.

Proposition 72. *Given $k \geq 1$ and $\phi(X_1, \ldots, X_n)$ an* MSO*-formula over Sig_k^m, $m \geq 0$, there exists an* MSO*-formula $\phi^{+1}(X_1, \ldots, X_n)$ over Sig_{k+1}^m such that*
$\forall R_1, \ldots, R_m \subseteq \mathcal{P}_k$, $\forall S_1, \ldots, S_n \in \mathcal{P}_{k+1}$,

$$\boldsymbol{\mathcal{P}_{k+1}}^{\mathrm{f}_k{}^{-1}(R_1, \ldots, R_m)} \models \phi^{+1}(S_1, \ldots, S_n) \text{ iff}$$

$\exists D_1, \ldots, D_n \subseteq \mathcal{P}_k$ *such that* $\boldsymbol{\mathcal{P}_k}^{(R_1, \ldots, R_m)} \models \phi(D_1, \ldots, D_n)$ *and* $\forall i$, $S_i = \mathrm{f}_k{}^{-1}(D_i)$.

PROOF. We proceed as in the proof of Corollary 60.

Proposition 73. *For every $k \geq 1$, for every $D \subseteq \mathcal{P}_k$ definable in $\boldsymbol{\mathcal{P}_k}$ the set ${\mathrm{f}_k}^{-1}(D)$ is* MSO*-definable in $\boldsymbol{\mathcal{P}_{k+1}}$.*

4.2.2. MSO*-properties of $\boldsymbol{\mathcal{P}_k}$*

We apply now transfer theorems to $\boldsymbol{\mathcal{P}_k}$. First remark that f_k is the restriction to $\mathcal{P}_{k+1}$ of the morphism $\mu_k : \widehat{A_{1,k+1}}^* \to \mathrm{Irr}(A_{1,k})$ mapping each $u \in \widehat{A_{1,k+1}}^*$ to $\rho(\pi_{\widehat{A_{1,k}}}(u))$ (recall that ρ is the reduction in the free group, see Section 1.1). Let us introduce for every $k \geq 1$, the vector $\vec{O}_k$ of subsets of $\mathcal{P}_k$ defined by the following induction:

- $\vec{O}_1 = \emptyset$
- $\vec{O}_{k+1} = ({\mathrm{f}_k}^{-1}(\vec{O}_k), {\mathrm{f}_k}^{-1}(\mathcal{P}_k a_1), \ldots, \mathrm{f}_k^{-1}(\mathcal{P}_k a_n))$ where $A_{1,k} = \{a_1, \ldots, a_n\}$

In other words, $\vec{O}_k$ consists in every ${\mathrm{f}_{k,i}}^{-1}(\mathcal{P}_i a)$, where $1 \leq i \leq k$ and $a \in A_{1,i}$.

Lemma 74. *For every $k \geq 1$, structures $\boldsymbol{\mathcal{P}_k}$, T_k, ${T_k}^{\vec{O}_k}$ and $\mathrm{f}_k({T_k}^{\vec{O}_k})$ are* MSO*-equivalent (see Definition 18).*

PROOF.
- Clearly, $\boldsymbol{\mathcal{P}_k}$ is definable inside T_k since $\forall u, v \in \mathcal{P}_k$, $a \in \widehat{A_{1,k}}$, $u = v \bullet a$ iff $u = va$ or $v = u\bar{a}$. Conversely, to show that T_k is definable inside $\boldsymbol{\mathcal{P}_k}$, we prove first that if $\widehat{A_{1,k}} = \{a_1, \ldots, a_n\}$, then the vector $(\mathcal{P}_k \cdot a_1, \ldots, \mathcal{P}_k \cdot a_n)$ is MSO-definable inside $\boldsymbol{\mathcal{P}_k}$. It suffices to remark that each $\mathcal{P}_k \cdot a_i$ is the smallest set $S_i \subseteq \mathcal{P}_k$ such that for every $u \in \mathcal{P}_k$, $u \in S_i$ iff
  - either $\bullet_{a_i}(u, \varepsilon)$,
  - or there exists $a_j \neq \overline{a_i} \in \widehat{A_{1,k}}$ and $v \in S_j$ such that $u = v \bullet a_i$.

  Now, it is easy to define in $\boldsymbol{\mathcal{P}_k}$ the relation induced by the concatenation product since for every $u, v \in \mathcal{P}_k$, $a \in \widehat{A_{1,k}}$, $[u = va$ iff $u = v \bullet a$ and $v \notin \mathcal{P}_k \cdot \overline{a}]$.
- ${T_k}^{\vec{O}_k}$ is definable inside $\boldsymbol{\mathcal{P}_k}$ since from the previous case, $\forall 1 \leq i \leq k$, $a \in A_{1,i}$, the set $\mathcal{P}_i \cdot a$ is MSO-definable in $\boldsymbol{\mathcal{P}_i}$ and then by using Proposition 73, ${\mathrm{f}_{k,i}}^{-1}(\mathcal{P}_i \cdot a)$ is MSO-definable in $\boldsymbol{\mathcal{P}_k}$.
- $\mathrm{f}_k({T_{k+1}}^{\vec{O}_{k+1}})$ is MSO-equivalent to the structure ${\boldsymbol{\mathcal{P}_k}}^{\vec{O}_k}$, which is MSO-equivalent to the structure $\boldsymbol{\mathcal{P}_k}$.

The following lemma is required to apply transfer theorems (see Definition 54).

Lemma 75. *For every $k \geq 1$, the property $\mathrm{TH}(\mathrm{f}_k, \vec{O_{k+1}})$ is satisfied.*

PROOF. It suffices to check that for every $k \geq 0$, $\mathcal{P}_{k+1} \in \mathsf{REG}^{\vec{O}_{k+1}}$. Consider $\mathcal{A}_{k+1} = (Q = \{q_0\} \cup \{q_a \mid a \in \widehat{A_{1,k+1}}\}, \widehat{A_{1,k+1}}, \Delta, q_0, F = Q)$ where Δ consists of every transitions

- $(q_0, a, \vec{o}, q_a)$, $\forall a \in A_{1,k+1}$, $\forall \vec{o}$
- $(q_b, a, \vec{o}, q_a)$, $\forall a \in A_{1,k+1}$, $\forall b \neq \bar{a}$, $\forall \vec{o}$
- $(q_b, \bar{a}, \vec{o}, q_a)$, $\forall a \in A_i$, $i \in [1,k]$ $\forall b \neq a$, $\forall \vec{o}$ such that the component corresponding to ${\mathrm{f}_{k,i}}^{-1}(\mathcal{P}_i a)$ is 1.

From Proposition 62, $\mathrm{L}(\mathcal{A}_{k+1}) = \mathcal{P}_{k+1}$.

Theorem 76. *For every $k \geq 1$, the structure $\boldsymbol{\mathcal{P}_k}$ has a decidable* MSO*-theory and fulfills the property* DM.

PROOF. We prove this result by induction on $k \geq 1$:

Basis: From Theorem 39, $\boldsymbol{\mathcal{P}_1}$ has a decidable MSO-theory and satisfies the property DM.

Induction step: let us suppose the property true for $k \geq 1$. Since $\mathrm{TH}(\mathrm{f}_k, \vec{O_{k+1}})$, by using Theorem 55 and equivalence between structures proved Proposition 74, the MSO-theory of $\boldsymbol{\mathcal{P}_{k+1}}$ is decidable. In the same way, since from Proposition 73, the map f_k is MSO-invertible, and by using Theorem 57, $\boldsymbol{\mathcal{P}_{k+1}}$ satisfies DM.

The decidability result has already been proved in [7].

The same kind of reasoning can be applied to the structure ${\boldsymbol{\mathcal{P}_k}}^{\vec{O}}$ for any vector $\vec{O}$ of subsets of $\mathcal{P}_k$: if the MSO-theory of ${\boldsymbol{\mathcal{P}_k}}^{\vec{O}}$ is decidable, then the MSO-theory of ${\boldsymbol{\mathcal{P}_{k+1}}}^{{\mathrm{f}_k}^{-1}(\vec{O})}$ is decidable.

Theorem 77. *Given $\vec{R}$ a vector of subsets of A_1^*, and $\vec{O} = {\mathrm{f}_{k,1}}^{-1}(\vec{R})$,*

1. *if the* MSO*-theory of $\langle {A_1}^*, \varepsilon, (succ_a)_{a \in A_1}, \vec{R}\rangle$ is decidable, then for every $k \geq 1$, the* MSO*-theory of ${\boldsymbol{\mathcal{P}_k}}^{\vec{O}}$ is decidable,*
2. *if $\langle {A_1}^*, \varepsilon, (succ_a)_{a \in A_1}, \vec{R}\rangle$ fulfills* DM*, then ${\boldsymbol{\mathcal{P}_k}}^{\vec{O}}$ fulfills* DM.

We define now the class FA_k of automata corresponding to languages MSO-definable in $\boldsymbol{\mathcal{P}_k}$ and the class REG_k of languages recognized by such automata.

Definition 78. For all $k \geq 1$, classes FA_k and REG_k are defined inductively as follows:

- FA_1 is the class of finite automata, and REG_1 the regular languages one,
- for every $k \geq 1$, FA_{k+1} consists in all automata $\mathcal{A}$ with p-oracle $({\mathrm{f}_k}^{-1}(R_1), \ldots, {\mathrm{f}_k}^{-1}(R_m))$ such that each R_i belongs to REG_k,

- for every $k \geq 1$, REG_{k+1} consists in all languages in $\mathcal{P}_k$ recognized by automata in FA_{k+1}.

Theorem 79. *For every language $L \subseteq \widehat{A_{1,k}}^*$ with $k \geq 1$, L is* MSO*-definable in $\boldsymbol{\mathcal{P}_k}$ iff L belongs to REG_k.*

Proof. Let us prove this result by induction on $k \geq 1$.

Basis: the case $k = 1$ is obvious,

Induction step: let us suppose the property is valid for $k \geq 1$.
From Theorem 57, any language $L \subseteq (k+1)$-pds is MSO-definable in $\boldsymbol{\mathcal{P}_{k+1}}$ iff there exists a vector $\vec{D}$ MSO-definable in $\boldsymbol{\mathcal{P}_k}$ and $\mathcal{A} \in \mathsf{FA}^{\vec{D}}$ such that $L = \mathrm{L}(\mathcal{A})$. By induction hypothesis, each component of $\vec{L}$ belongs to REG_k, and then, L is MSO-definable in $\boldsymbol{\mathcal{P}_{k+1}}$ iff $D \in \mathsf{REG}_{k+1}$.

4.3. Regular sets of k-pushdowns

We now translate results obtained on $\mathcal{P}_k$ in terms of k-pushdowns. For that, we just need to prove that $\boldsymbol{\mathcal{P}_k}$ and the structure PDS_k associated to the type k-pushdowns (see Section 1.5.1(3)) are MSO-equivalent (see Definition 18) . We show that $\varphi_k : \mathsf{PDS}_k \to \boldsymbol{\mathcal{P}_k}$ and ${\varphi_k}^{-1} : \boldsymbol{\mathcal{P}_k} \to \mathsf{PDS}_k$ are MSO-interpretations. Then, the two structures have the same MSO-properties and we have a nice class of p-oracle-automata available to characterize the class of all $\varphi_k(D)$ such that D is MSO-definable in PDS_k. Using this automata characterization, we define the class of controlled k-pds systems of transitions generating the class of all sets MSO-definable in PDS_k.

Theorem 80. *For every $k \geq 1$, ${\varphi_k}^{-1} : \boldsymbol{\mathcal{P}_k}(A_1, \ldots, A_k) \to \mathsf{PDS}_k(A_1, \ldots, A_k)$ is a* MSO*-interpretation.*

Proof. Let us check that conditions of the definition of MSO-interpretation (Definition 16) are well satisfied,

1. ${\varphi_k}^{-1}(\mathcal{P}_k) = k$-$\mathsf{pds}$ is MSO-definable in PDS_k
2. from the Lemma 68, it follows that for every $u, v \in \mathcal{P}_k$, $a \in A_i$, $1 \leq i \leq k$:
   $\boldsymbol{\mathcal{P}_k} \models \bullet_a(u, v)$ iff $\mathsf{PDS}_k \models \mathrm{push}_a({\varphi_k}^{-1}(u), {\varphi_k}^{-1}(v))$ and
   $\boldsymbol{\mathcal{P}_k} \models \bullet_{\bar{a}}(u, v)$ iff $\mathsf{PDS}_k \models \mathrm{push}_a({\varphi_k}^{-1}(v), {\varphi_k}^{-1}(u))$.

Let us prove now that $\forall k \geq 1$, $\varphi_k : \mathsf{PDS}_k \to \boldsymbol{\mathcal{P}_k}$ is a MSO-interpretation. The next lemma establishes that to prove that an instruction of level k is MSO-definable in $\boldsymbol{\mathcal{P}_{k+i}}$, $i \geq 0$, there is only to need to demonstrate that it is MSO-definable in $\boldsymbol{\mathcal{P}_k}$:

Lemma 81. *Given instr an instruction of level $k \geq 1$, and $\Phi(x, y)$ a* MSO*-formula over Sig_k satisfying for all $u, v \in \mathcal{P}_k$:*

$$\boldsymbol{\mathcal{P}_k} \models \Phi(u, v) \text{ iff } {\varphi_k}^{-1}(v) = \mathit{instr}({\varphi_k}^{-1}(u)),$$

then for every $i \geq 0$, *there exists a formula* $\Phi_{+i}(x,y) \in \mathrm{MSO}(Sig_{k+i})$ *such that* $\forall u, v \in \mathcal{P}_{k+i}$,

$$\boldsymbol{\mathcal{P}_{k+i}} \models \Phi_{+i}(u,v) \text{ iff } {\varphi_{k+i}}^{-1}(v) = \mathsf{instr}({\varphi_{k+i}}^{-1}(u)).$$

Proof. From proposition 65 and definition of ${\varphi_k}^{-1}$, it follows that for every $v, v' \in (k+1)$-pds and $\omega_1, \omega_1' \in k$-pds, the following properties are equivalent:

1. there exists $\omega \in (k+1)\text{-pds} \cup \{\varepsilon\}$ and $a \in A_{k+1} \cup \{\bot\}$ such that
   ${\varphi_{k+1}}^{-1}(v) = a[\omega_1]\omega$ and ${\varphi_{k+1}}^{-1}(v') = a[\omega_1']\omega$
2. there exists $u \in \mathrm{Irr}(A_k)$ such that $v' = v \bullet u$ and
   $f_k(v) = \varphi_k(\omega_1)$ and $f_k(v') = \varphi_k(\omega_1')$.

Then, given instr an instruction of level k and Φ MSO-defining instr in $\boldsymbol{\mathcal{P}_k}$, we obtain (by using formulas constructed in Proposition 72) the following iterative construction of Φ_{+i}:

$$\Phi_{+0}(x,y) := \Phi(x,y)$$

$$\forall i \geq 0,\ \Phi_{+(i+1)}(x,y) := \exists u,\ y = x \bullet u \wedge (\Phi_{+i})^{+1}(x,y).$$

Theorem 82. *For every* $k \geq 1$, $\varphi_k : \mathsf{PDS}_k \to \boldsymbol{\mathcal{P}_k}$ *is a* MSO*-interpretation.*

Proof. By using the previous lemma, it only remains to show that $\mathrm{push}_{k,a}$, pop_k and $\mathrm{change}_{k,a}$ are MSO-definable in $\boldsymbol{\mathcal{P}_k}$:
for every $a \in A_i$, $1 \leq i \leq k$,

$$\textsc{push}_{k,a}(x,y) := \ y = x \bullet a$$

$$\textsc{pop}_{k,a}(x,y) := \exists a \in A_k,\ \exists w,\ x = y \bullet a \bullet w$$

$$\textsc{change}_{k,a}(x,y) := \exists u, u',\ \exists b \in A_k,\ y = x \bullet u' \bullet \bar{b} \bullet a \bullet u \wedge x\,(=)^{+1}\,y$$

Corollary 83. *For every* $k \geq 1$, *every* $D \subseteq k$-pds,
D *is* MSO*-definable in* PDS_k *iff* $\varphi_k(D)$ *is* MSO*-definable in* $\boldsymbol{\mathcal{P}_k}$.

We now translate the properties of k-regular languages in terms of k-pushdowns by using the MSO-equivalence of the structures $\boldsymbol{\mathcal{P}_k}$ and PDS_k.
The following theorem is straightforward from Theorem 76.

Theorem 84. *For every* $k \geq 1$, *the structure* PDS_k *has a decidable* MSO*-theory and fulfills the property* DM.

The decidability result is proved in [19] by using Muchnik's Theorem (see [27] or [35]). Finally, we show that REG^k admits several characterizations that extend the REG ones.

Theorem 85. *For every* $S \subseteq k\text{-pds}(A_1, \ldots, A_k)$, $k \geq 1$, *the following properties are equivalent:*

1. S *is generated by a* k*-pds system of transitions whose controller are* MSO*-definable in* $\mathsf{PDS}_k(A_1, \ldots, A_k)$

2. S *is* MSO-*definable in* $\mathsf{PDS}_k(A_1, \ldots, A_k)$
3. $\varphi_k(S)$ *is* MSO-*definable in* $\boldsymbol{\mathcal{P}_k}(A_1, \ldots, A_k)$
4. $\varphi_k(S)$ *is recognized by an automaton in* $\mathsf{FA}_k(A_1, \ldots, A_k)$.

Proof. Equivalence between 2 and 3 stems from the equivalence between the two structures. Equivalence between 3 and 4 is established in Theorem 76.
Given a k-pds system of transitions $\mathcal{S}$ controlled by a vector $\vec{C}$ of sets which are MSO-definable in PDS_k. It is possible to write a formula defining in PDS_k the set of k-pds generated by $\mathcal{S}$. So 1 implies 2.
Let us end the proof by showing that 4 implies 1. Given $k \geq 0$ and $\mathcal{A} = (Q, \widehat{A_{1,k}}, \vec{R}, \Delta, q_0, F) \in \mathsf{FA}_k$, we are going to construct $\mathfrak{A} \in k\text{-}\mathsf{TS}^{\vec{C}}$ with $\vec{C} = {\varphi_k}^{-1}(\vec{R})$ fulfilling $\varphi_k(\mathrm{L}(\mathcal{A})) = \mathrm{P}(\mathfrak{A})$.
From the equivalence between 2 and 3, $\vec{C}$ is a vector of sets which are MSO-definable in PDS_k. We can suppose w.l.o.g. that $\mathcal{A}$ is complete in $\mathcal{P}_k$, i.e., that $\forall u \in {\widehat{A_{1,k}}}^*$, u is computed by $\mathcal{A}$ iff $u \in \mathcal{P}_k$.
Let us set $\mathfrak{A} = (Q, (A_1, \ldots, A_k), \Delta', \vec{C}, q_0, \bot, F)$ where Δ' is constructed in the following way:

- $\forall (p, a, \vec{o}, q) \in \Delta$, $a \in A_i$, $1 \leq i \leq k+1$, then $\forall w \in \mathrm{top}(k\text{-}\mathsf{pds}(A_1, \ldots, A_k))$
$$(p, w, \vec{o}, \mathrm{push}_{i,a}, q) \in \Delta'$$
- $\forall (p, \bar{a}, \vec{o}, q) \in \Delta$, $a \in A_i$, $1 \leq i \leq k$, then $\forall w \in \mathrm{top}(k\text{-}\mathsf{pds}(A_1, \ldots, A_k))$
$$(p, w, \vec{o}, \mathrm{pop}_i, q) \in \Delta'.$$

It can be easily checked that $\varphi_k(\mathrm{L}(\mathcal{A})) = \mathrm{P}(\mathfrak{A})$.

Remark 86.

1. It can be proved that languages recognized by k-pds automata controlled by MSO-definable sets are languages recognized by k-pds automata without controllers.
2. The equivalence between (1) and (4) is proved in [21] for $k = 1$.

5. Final comment

The work presented here is a part of the author's PhD presented at LaBRI, Bordeaux University on the theme of Iterated Pushdown automata [18]. It is shown there that Theorem 85 has several applications.

For example, by using the automata characterization of REG_k, it can be proved that the projection in 1-pds of a pushdown set generated by a k-pds system of transitions is regular. This result allows the comparison between the two classes of predicates P given in [19] and [8] for which the MSO-theory of $\langle \mathbb{N}, +1, P \rangle$ is decidable. It can be proved that all sequences of level k are *profinitely ultimately periodic* and the class of predicates describes in [19] is then included in the one described in [8].

Theorem 85 also allows to define a large class of tuples $(P_1, \ldots, P_m)$ of unary predicates for which the MSO-theory of $\langle \mathbb{N}, +1, P_1, \ldots, P_m \rangle$ is decidable for [17].

Recent work deals with the notion of regular sets of "higher-order pushdowns" (hop) which are restricted *it*-pushdowns. In [3], a set S of k-hops is called *regular* if the set of words in $(\mathcal{A}_k \cup \{[,]\})^*$ representing S is accepted by a finite automaton. It is shown that for any higher-order process with a single state, the set of all predecessors of a given *regular* set of configurations is *regular*.

In [6], the author introduces a notion of regular sets of higher-order pushdowns (hop). He studies the classe Reg_k corresponding to the sets of k-hop accessible by using only instruction push and $\overline{\text{push}}$. He gives a normalized representation of this class using regular expressions over a monoïde in $(\widehat{A_{1,k}} \cup T_k)^*$, where T_k is an infinite alphabet consisting of all symbols T_R, for $R \in Reg_{k-1}$. This normalization extended the one obtained in [9] for the level 1. The author proves also that the class Reg_k corresponds to the class of sets MSO-definable in PDS_k. The class Reg_k correspond then to the image by φ_k of the class REG^k defined in the previous section. These two disjoints works prove that the class REG^k enjoyes numerous properties generalizing the PREG_1 ones (which correspond by isomorphism to the class REG). In addition, the representation of k-pds by word in the free group seems to be very well adapted to the generalization of the notion of regular languages.

Acknoledgements

References

[1] A. V. Aho. Indexed grammars—an extension of context-free grammars. *J. Assoc. Comput. Mach.*, 15:647–671, 1968.

[2] A. V. Aho. Nested stack automata. *J. Assoc. Comput. Mach.*, 16:383–406, 1969.

[3] A. Bouajjani and A. Meyer. Symbolic reachability analysis of higher-order context-free processes. In *FSTTCS 2004: Foundations of software technology and theoretical computer science*, volume 3328 of *Lecture Notes in Comput. Sci.*, pages 135–147. Springer, Berlin, 2004.

[4] J.R. Büchi. Weak second-order arithmetic and finite automata. *Z. Math. Logik Grundlagen Math.*, 6:66–92, 1960.

[5] T. Cachat. Higher order pushdown automata, the Caucal hierarchy of graphs and parity games. In *Automata, languages and programming*, volume 2719 of *Lecture Notes in Comput. Sci.*, pages 556–569. Springer, Berlin, 2003.

[6] A. Carayol. Regular sets of higher-order pushdown stacks. In *MFCS*, volume 3618 of *Lecture Notes in Comput. Sci.*, pages 168–179. Springer, 2005.

[7] A. Carayol and S. Wöhrle. The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In *FST TCS 2003: Foundations of software technology and theoretical computer science*, volume 2914 of *Lecture Notes in Comput. Sci.*, pages 112–123. Springer, Berlin, 2003.

[8] O. Carton and W. Thomas. The monadic theory of morphic infinite words and generalizations. *Inform. and Comput.*, 176(1):51–65, 2002.

[9] D. Caucal. On infinite transition graphs having a decidable monadic theory. In *Automata, languages and programming (Paderborn, 1996)*, volume 1099 of *Lecture Notes in Comput. Sci.*, pages 194–205. Springer, Berlin, 1996.

[10] B. Courcelle. Fundamental properties of infinite trees. *Theoret. Comput. Sci.*, 25(2):95–169, 1983.

[11] B. Courcelle. The monadic second-order logic of graphs. IX. Machines and their behaviours. *Theoret. Comput. Sci.*, 151(1):125–162, 1995. Topology and completion in semantics (Chartres, 1993).

[12] B. Courcelle and I. Walukiewicz. Monadic second-order logic, graph coverings and unfoldings of transition systems. *Ann. Pure Appl. Logic*, 92(1):35–62, 1998.

[13] W. Damm and A. Goerdt. An automata-theoretical characterization of the OI-hierarchy. *Inform. and Control*, 71(1-2):1–32, 1986.

[14] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *Proceedings of the 32nd annual symposium on Foundations of computer science*, pages 368–377, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.

[15] J. Engelfriet. Iterated pushdown automata and complexity classes. In *Proceedings of the 14th Symposium on Theory of Computing*, pages 365–373. Association for Computing Machinery, 1983.

[16] J. Engelfriet. Iterated stack automata and complexity classes. *Inform. and Comput.*, 95(1):21–75, 1991.

[17] S Fratani. The theory of successor extended with several predicates. Available at url: http://www.cmi.univ-mrs.fr/~sfratani/.

[18] S Fratani. *Automates à piles de piles ... de piles.* PhD thesis, Université Bordeaux 1, 2005.

[19] S. Fratani and G. Sénizergues. Iterated pushdown automata and sequences of rational numbers. In *Ann. Pure Appl. Logic*, volume 141, pages 363–411. Elsevier, 2005.

[20] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Comput. Sci.* Springer, 2002.

[21] S. A. Greibach. A note on pushdown store automata and regular systems. *Proc. Amer. Math. Soc.*, 18:263–268, 1967.

[22] S. A. Greibach. Full AFLs and nested iterated substitution. *Inform. and Control*, 16:7–35, 1970.

[23] Y. Gurevich and L. Harrington. Trees, automata, and games. In *Proceedings of the 14th Symposium on Theory of Computing*, pages 60–65. Association for Computing Machinery, 1982.

[24] T. Knapik, D. Niwiński, and P. Urzyczyn. Higher-order pushdown trees are easy. In *Foundations of software science and computation structures (Grenoble, 2002)*, volume 2303 of *Lecture Notes in Comput. Sci.*, pages 205–222. Springer, Berlin, 2002.

[25] A. N. Maslov. The hierarchy of index languages of arbitrary level. *Dokl. Akad. Nauk SSSR*, 217:1013–1016, 1974.

[26] A. N. Maslov. Multilevel pushdown automata. *Problemy Peredači Informacii*, 12(1):55–62, 1976.

[27] A. Muchnik and A.L. Semenov. Automata on infinite objects, monadic theories, and complexity. In *Dagstuhl-Seminar*, 1992.

[28] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141:1–35, 1969.

[29] M.O. Rabin. Decidable theories. *Handbook of Mathematical Logic*, pages 595–629, 1977.

[30] A. Rabinovich. On decidability of monadic logic of order over the naturals extended by monadic predicates. *Information and Computation*, 205(6):870–889, 2007.

[31] A. L. Semenov. Decidability of monadic theories. In *Mathematical foundations of computer science, 1984 (Prague, 1984)*, volume 176 of *Lecture Notes in Comput. Sci.*, pages 162–175. Springer, Berlin, 1984.

[32] W. Thomas. The theory of successor with an extra predicate. *Math. Ann.*, 237(2):121–132, 1978.

[33] W. Thomas. Automata on infinite objects. In *Handbook of theoretical computer science, Vol. B*, pages 133–191. Elsevier, Amsterdam, 1990.

[34] W. Thomas. Constructing infinite graphs with a decidable MSO-theory. In *Mathematical foundations of computer science 2003*, volume 2747 of *Lecture Notes in Comput. Sci.*, pages 113–124. Springer, Berlin, 2003.

[35] I. Walukiewicz. Monadic second-order logic on tree-like structures. *Theoret. Comput. Sci.*, 275(1-2):311–346, 2002.